VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Krokoměr s mikropočítačem ARM

Pedometer with ARM Microcomputer

2019 Filip Šigut

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Filip Šigut**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Krokoměr s mikropočítačem ARM
Pedometer with ARM Microcomputer

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je realizovat krokoměr pomocí běžně dostupného vývojového KITu s procesorem ARM. Pro realizaci práce je potřeba vybrat a otestovat vhodný polohový senzor a navrhnout a vyzkoušet vhodné algoritmy pro filtraci získaných dat a detekci kroků.

1. Seznamte se s polohovými senzory a vyberte minimálně 3 různé čipy.
2. Připojte vybrané senzory k mikropočítači a zaznamenejte data při chůzi, minimálně desítky minut.
3. Vyzkoušejte vhodnost několika filtrů digitálních signálů pro naměřená data a vyhodnoťte jejich vhodnost pro krokoměr.
4. Pro filtrovaná data navrhněte a vyzkoušejte vhodné možnosti detekce kroků.
5. Otestované algoritmy přeneste do mikropočítače a otestujte funkcionalitu.
6. Navrhněte jednoduché programové rozhraní pro komunikaci mikropočítače a počítače, aby bylo možno z mikropočítače získat potřebná data.
7. Otestujte zařízení a porovnejte výsledky s libovolným dostupným krokoměrem.

Seznam doporučené odborné literatury:

[1] Mikropočítač K64F a vývojový kit FRDM-K64F, http://www.nxp.com
[2] Skapa Jan, Algoritmy a systémy digitálního zpracování signálů pro integrovanou výuku VUT a VŠB-TUO, skripta, 2014, https://vut-vsb.cz/home/get-file?file=459&portal=Portal2
[3] Weinberg Harvey, Using the ADXL202 in Pedometer and Personal Navigation Applications, application note AN-206, Analog device, http://www.bdtic.com/DownLoad/ADI/AN-602.pdf
[4] Sheu, J.S.; Huang, G.S.; Jheng, W.C.; Hsiao, C.H. Design and Implementation of a Three-Dimensional Pedometer Accumulating Walking or Jogging Motions. In Proceedings of the 2014 International Symposium on Computer, Consumer and Control (IS3C), Taichuang, Taiwan, 10–12 June 2014; pp. 828–831.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Petr Olivka, Ph.D.**

Datum zadání: 01.09.2017

Datum odevzdání: 30.04.2019

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 30. dubna 2019 .....

Rád bych na tomto místě poděkoval svému vedoucímu bakalářské práce Ing. Petrovi Olivkovi, Ph.D. za jeho odbornou pomoc a trpělivost při řešení této prace.

Abstrakt

Tato bakalářská práce se zabývá implementací krokoměru na vývojové desce NXP FRDM-K64F. Pro zaznamenání chůze jsou vybrány a porovnány různé MEMS akcelerometry. Pro odstranění šumu z naměřeného signálu je otestována řada různých filtrů. Dále jsou implementovány a na filtrovaném signálu otestovány algoritmy detekce chůze. Výsledky těchto měření jsou porovnány s výsledky komerčně dostupných krokoměru. Kombinace vhodného filtru a způsobu detekce kroků je následně implementována na vývojovou desku NXP FRDM-K64F.

Klíčová slova: krokoměr, akcelerometr, NXP, FRDM-K64F

Abstract

This bachelor thesis deals with the implementation of a pedometer on the NXP FRDM-K64F development board. Various MEMS accelerometers are selected and compared to record walking. A variety of filters are tested to remove noise from the measured signal. Furthermore, step detection algorithms are implemented and tested on the filtered signal. The results of these measurements are compared with commercially available pedometers. The combination of a suitable filter and a method of step detection is then implemented on the NXP FRDM-K64F development board.

Key Words: pedometer, accelerometer, NXP, FRDM-K64F

Obsah

Seznam použitých zkratek a symbolů

MCU	–	Microcontroller unit
SPI	–	Serial Peripheral Interface
I^2C	–	Inter-Integrated Circuit
SCL	–	Serial Clock
SDA	–	Serial Data
ACK	–	Acknowledge
NACK	–	Not Acknowledge
NVIC	–	Nested Vectored Interrupt Controller
MSB	–	Most significant byte
LSB	–	Least significant byte
GPIO	–	General-purpose input/output
MEMS	–	Microelectromechanical systems
USB	–	Universal Serial Bus
SD	–	Secure Digital
SDK	–	Software development kit
IDE	–	Integrated development environment

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Cílem této bakalářské práce je vytvořit krokoměr pomocí vývojového kitu s ARM procesorem.

Krokoměr je zařízení, které počítá kroky tím, že detekuje vertikální pohyb těla při chůzi [1][2]. Krokoměry jsou v dnešní době velmi populárním způsobem jak motivovat lidi k lepšímu životnímu stylu a jsou obvykle nošeny v kapse či ve formě náramku na ruce.

V této práci budou vybrány a otestovány vhodné senzory schopny zaznamenat chůzi. Data důležitá pro tuto práci budou naměřena za doprovodu dvou volně dostupných krokoměrů pro porovnání výsledků. Bude zde popsán způsob přenosu a ukládání naměřených dat. Následně budou naměřená data zpracována na počítači. Pro snazší detekci chůze bude otestováno a porovnáno několik filtrů pro redukci šumu aplikovaných na signál naměřený těmito senzory. Budou také otestovány a popsány způsoby detekce kroků. Po výběru vhodného filtru a algoritmu pro detekci kroků budou tyto algoritmy implementovány na vývojový kit s ARM procesorem.

2 Výběr vývojové desky

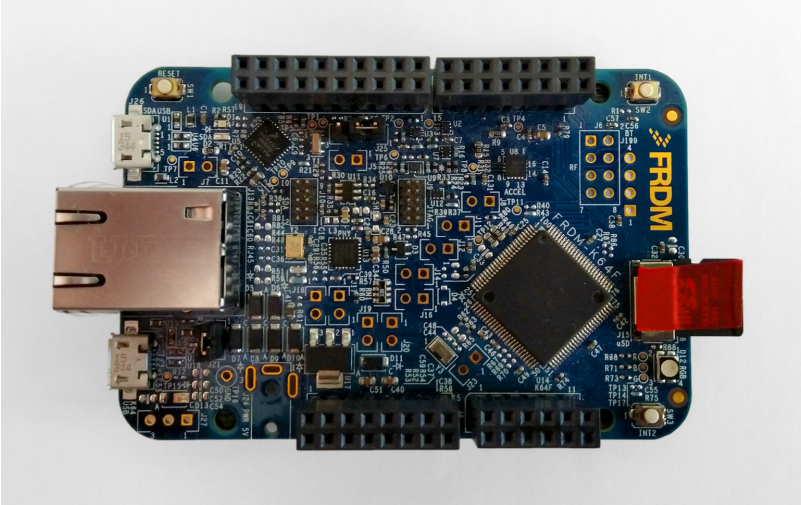
Kvůli tomu, že budou probíhat měření s vývojovou deskou v terénu, je důležité vybrat takovou, kterou lze jednoduše napájet, a která umožňuje ukládání dat pro jejich pozdější zpracování.

Jako vhodná byla vybrána vývojová deska NXP FRDM-K64F. Tuto desku lze napájet připojeným micro USB kabelem. Pro připojení externích senzorů lze využít I/O konektory. Dále je tato deska vybavena slotem pro micro SD kartu, na kterou lze ukládat naměřená data. Další výhodou je to, že tato deska je vybavena akcelerometrem. Tento senzor totiž umožňuje rozpoznání chůze, viz sekce 3.

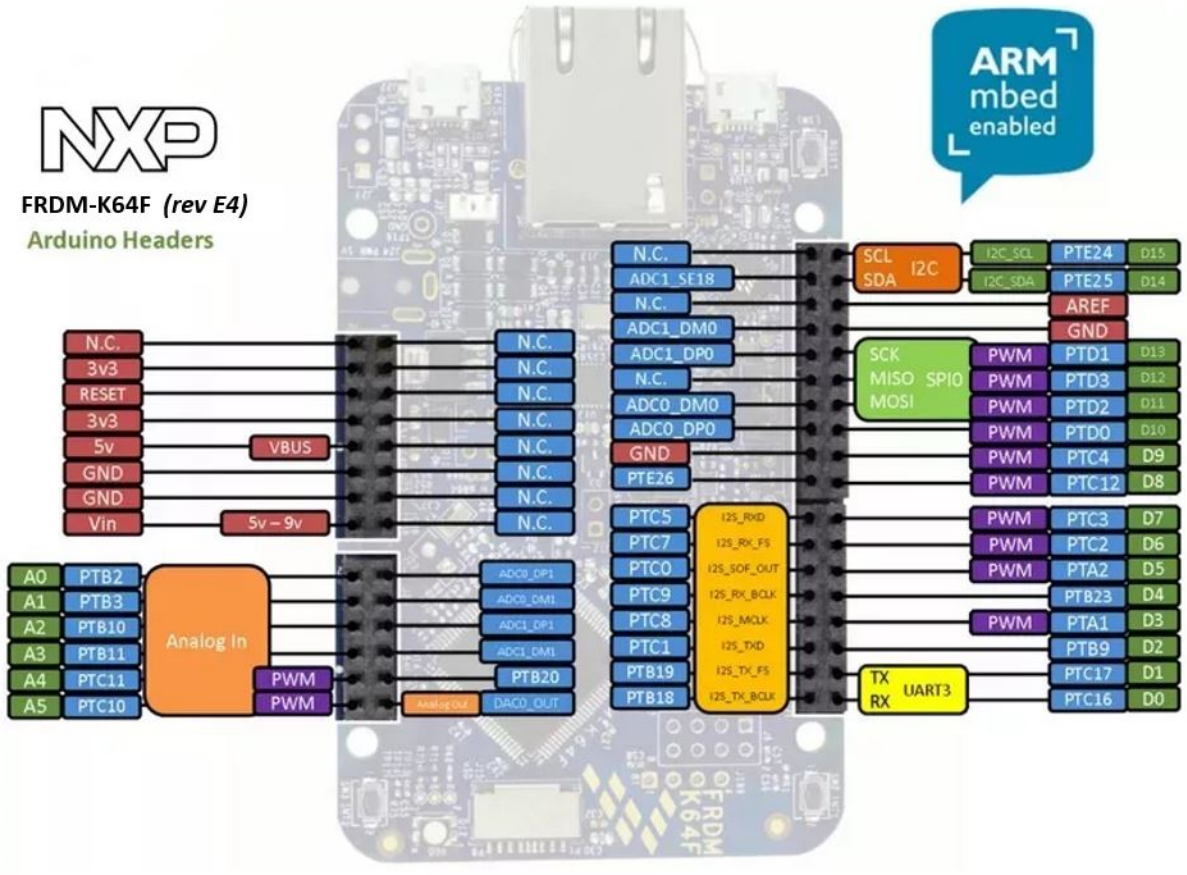
2.1 Vývojová deska NXP FRDM-K64F

NXP FRDM-K64F, viz obrázek 1, je vývojová deska postavená na ARM Cortex-M4 jádru, obsahující Kinetis® K Series mikrokontrolér MK64FN1M0VLL12 v 100LQFP pouzdře běžící na frekvenci až 120 MHz. Deska je vybavena Arduino R3 kompatibilními I/O konektory, slotem pro micro SD kartu, Ethernet portem, RGB LED, dvěma tlačítky a dvěma micro USB porty, z nichž jeden je připojen k integrovanému obvodu určenému k nahrání a ladění programů, a druhý je připojen k samotnému mikrokontroléru [3].

Programování a ladění cílové aplikace probíhá v MCUXpresso IDE ve verzi 10.3.0 za použití MCUXpresso SDK pro desku NXP FRDM-K64F ve verzi 2.5.0.



Obrázek 1: Vývojová deska NXP FRDM-K64F



Obrázek 2: Vývojová deska NXP FRDM-K64F s pojmenováním pinů [3]

3 Výběr vhodných senzorů

Hlavním požadavkem pro výběr vhodného senzoru je samozřejmě vybrat senzor, který bude schopen zachytit chůzi. V dnešní době většina krokoměrů spoléhá na MEMS akcelerometry a na algoritmy detekce kroků. Touto cestou se vydává i tato práce.

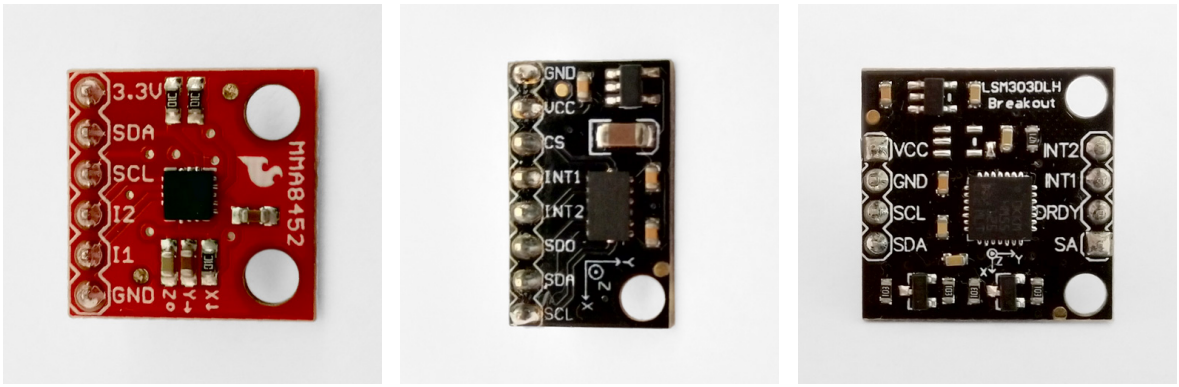
Akcelerometr je zařízení měřící zrychlení a, jehož jednotkou je ms^{-2}, které na něj působí. Kroky je tedy potřeba dělat na povrchu nějakého kosmického tělesa, což je v našem případě Země. Například akcelerometr v klidu na povrchu Země by naměřil velikost zrychlení odpovídající $g \approx 9.81\ ms^{-2}$. Na druhou stranu akcelerometr spuštěn do volného pádu ve vakuu by naměřil $g = 0\ ms^{-2}$.

Způsob jakým lze detekovat chůzi pomocí akcelerometru je ten, že při každém kroku vzniká vertikální pohyb těla nahoru a dolu. Tento pohyb je akcelerometrem zaznamenán jako změna velikosti tíhového zrychlení g [1][2].

3.1 Výběr akcelerometrů

Při výběru akcelerometrů jsou preferovány senzory již připájené na nějakém plošném spoji s vyvedenými kontakty pro snadné zapojení. Rozhodující je také cena, neboť se tyto senzory pohybují od částky sta korun do částek převyšující tisíce korun. Důležité také je vybrat akcelerometry schopné komunikace prostřednictvím I^2C nebo SPI rozhraní. Při výběru vhodného akcelerometru také záleží na počtu os. Preferovány jsou pouze tříosé akcelerometry, a to z důvodů, že u nich nezáleží v jaké pozici budou při měření umístěny.

Vývojová deska NXP FRDM-K64F již obsahuje jeden akcelerometr, a to NXP FXOS8700CQ. Společně s tímto akcelerometrem byly vybrány 3 další, a to NXP MMA8452Q, Analog Devices ADXL345 a STMicroelectronics LSM303DLH. Všechny tyto akcelerometry podporují komunikaci prostřednictvím I^2C rozhraní. Všechny vybrané moduly s akcelerometry jsou schopny provozu při napájení 3,3 voltů.



(a) NXP MMA8452Q (b) Analog Devices ADXL345 (c) STMicroelectronics LSM303DLH

Obrázek 3: Vybrané akcelerometry

3.1.1 NXP FXOS8700CQ

Akcelerometr NXP FXOS8700CQ je součástí desky NXP FRDM-K64F. Jedná se o tříosý akcelerometr a tříosý magnetometr. Podporuje jak I^2C, tak i SPI rozhraní. Má volitelný rozsah $\pm 2g/\pm 4g/\pm 8g$ a 14bitové rozlišení ve všech rozsazích. Má dva nastavitelné piny pro signalizaci přerušení a je schopen odesílat data rychlostí až 800 vzorků za sekundu [4].

3.1.2 NXP MMA8452Q

Akcelerometr NXP MMA8452Q je součástí modulu SPARKFUN ELECTRONICS INC. BOB-13926, viz obrázek 3(a). Jedná se o tříosý akcelerometr a je schopen komunikace pouze přes I^2C rozhraní. Má volitelný rozsah $\pm 2g/\pm 4g/\pm 8g$ a 12bitové rozlišení ve všech rozsazích. Má dva nastavitelné piny pro signalizaci přerušení a je schopen odesílat data rychlostí až 800 vzorků za sekundu [5].

3.1.3 Analog Devices ADXL345

Akcelerometr Analog Devices ADXL345 je součástí modulu DFROBOT SEN0032, viz obrázek 3(b). Jedná se o tříosý akcelerometr a je schopen komunikace jak přes I^2C, tak i SPI rozhraní. Má volitelný rozsah $\pm 2g/\pm 4g/\pm 8g/\pm 16g$. Rozlišení je závislé na zvoleném rozsahu, a to 10 bitů u $\pm 2g$ až 13 bitů u $\pm 16g$. Má dva nastavitelné piny pro signalizaci přerušení. Maximální rychlost vzorkování je závislá na zvoleném rozhraní. V případě I^2C lze maximálně dosáhnout rychlosti odesílání dat 800 vzorků za sekundu [6].

3.1.4 STMicroelectronics LSM303DLH

Akcelerometr STMicroelectronics LSM303DLH je součástí modulu DFROBOT SEN0079, viz obrázek 3(c). Jedná se o tříosý akcelerometr a tříosý magnetometr. Podporuje pouze komunikaci prostřednictvím I^2C rozhraní. Má volitelný rozsah $\pm 2g/\pm 4g/\pm 8g$ a 12bitové rozlišení ve všech rozsazích. Má dva nastavitelné piny pro signalizaci přerušení a je schopen posílat data rychlostí až 1000 vzorků za sekundu [7].

4 Komunikace s akcelerometry

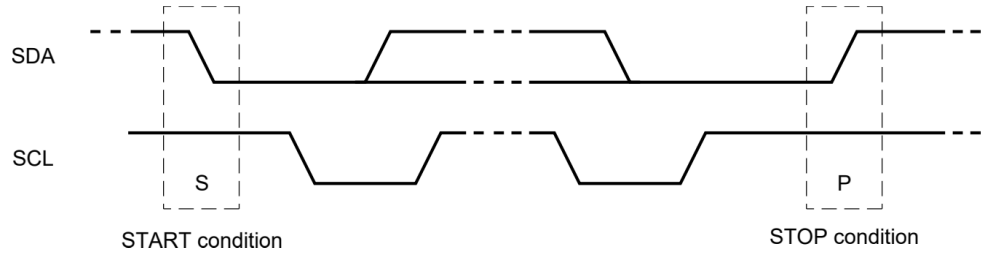
Komunikace s vybranými senzory probíhá po I^2C sběrnici. Výhodou této sběrnice je, že všechna zařízení připojena na tuto sběrnici jsou adresovatelná. Tohoto bylo využito při implementaci krokoměru na desce NXP FRDM-K64F, neboť toto řešení umožňuje automaticky najít a identifikovat připojené senzory.

4.1 Komunikace po I^2C sběrnici

I^2C sběrnice jsou dva vodiče představující dva signály: hodinový signál SCL a datový signál SDA. Hodinový signál SCL je vždy generován zařízením v roli Master. Datový signál SDA je používán pro přenos dat mezi zařízeními. Zařízení v roli Slave jsou na sběrnici identifikována pomocí 7bitové adresy. Tyto adresy jsou unikátní pro každé zařízení připojené na tuto sběrnici. Přenos dat na sběrnici vždy zahajuje Master. Oba vodiče SCL a SDA jsou připojeny pull-up rezistorem k V_{CC}. To znamená, že v klidovém stavu jsou oba vodiče ve stavu logická 1 a je na připojených zařízeních aby SCL a SDA připojovaly ke GND pro logickou 0.

Přenos na sběrnici začíná START sekvencí a končí STOP sekvencí, viz obrázek 4. Ty vždy generuje Master. START sekvence je definována jako přechod SDA ze stavu 1 na 0, zatímco je SCL ve stavu 1. STOP sekvence je definována jako přechod SDA ze stavu 0 na 1, zatímco je SCL ve stavu 1. Sběrnice je považována za obsazenou po vyslání START sekvence, a je tak až do vyslání STOP sekvence, kdy je sběrnice opět považována za volnou. Při normálním přenosu dat se SDA může měnit jen pokud je SCL ve stavu 0.

Při přenosu dat vysílač vždy posílá osm bitů, tedy jeden bajt. Každý přenesený bit je řízen hodinovým signálem na SCL generovaný Masterem. Po odeslání jednoho bajtu přijímač potvrdí přijatý bajt odesláním ACK (acknowledge) bitu, a to logickou 0. V případě kdy SDA není logická 0, vysílač chápe tuto situaci jako NACK (not acknowledge), tedy nepotvrzeno [13].



Obrázek 4: START a STOP sekvence [13]

4.1.1 Zápis dat

K zápisu dat Master pošle na I^2C sběrnici START sekvenci následovanou bajtem s horními sedmi bity určujícími adresu zařízení a $R/\overline{W}$ bitem 0 pro zápis. Poté co Slave odešle ACK bit, Master pošle adresu registru do kterého chce zapisovat a Slave opět potvrzuje odesláním ACK bitu.

Master následně posílá bajty, které Slave potvrzuje odesláním ACK bitu. Master po ukončení přenosu posílá STOP sekvenci [13].

4.1.2 Čtení dat

Čtení je podobné zápisu. Master pošle na I^2C sběrnici START sekvenci následovanou bajtem s horními sedmi bity určujícími adresu zařízení a R/$\overline{W}$ bitem 0 pro zápis. Poté co Slave odešle ACK bit, Master pošle adresu registru ze kterého chce číst a Slave potvrzuje odesláním ACK bitu.

Následně Master znovu posílá START sekvenci následovanou bajtem s horními sedmi bity určujícími adresu zařízení a R/$\overline{W}$ bitem 1 pro čtení. Slave odesílá ACK bit. Master uvolní SDA, ale stále generuje hodinový signál na SCL. Slave se při odesílání dat řídí hodinovým signálem, který je generován Masterem. V tomto okamžiku je Master přijímačem a přijímá odeslané bajty, které Master následně potvrzuje odesláním ACK bitu. Při příjmu posledního bajtu Master odesílá NACK bit následovaný STOP sekvencí [13].

4.2 Identifikace akcelerometrů

Byla implementována schopnost vyhledat a identifikovat připojené akcelerometry. Implementace vyhledání je ve výpisu 1. Toto umožňuje při spuštění krokoměru jejich identifikování a správné nastavení. Pro identifikaci připojených akcelerometrů, byla využita vlastnost I^2C sběrnice, kdy jde zjistit jaké adresy jsou na sběrnici již připojeny. K identifikaci jednotlivých akcelerometrů je potřeba, aby každý z nich měl jinou adresu. Naštěstí všechny vybrané akcelerometry bylo možné nastavit tak, aby každý byl identifikován svou vlastní adresou. Nastavené adresy jsou vypsány v tabulce 1.

```
for (int i = 0; i < sizeof(accel_addresses)/sizeof(accel_addresses[0]); i++)
{
    if(BOARD_I2C_Send(I2C0, accel_addresses[i], 0, 0, 0, 0) == kStatus_Success)
        {
        accel_address = accel_addresses[i];
        break;
    }
}
```

Výpis 1: Vyhledání připojeného akcelerometru

Kvůli tomu, že NXP FXOS8700CQ je již součásti vývojové desky a je vždy připojen k dané I^2C sběrnici, tak je potřeba nejprve zkontrolovat, zda není připojen nějaký z externích akcelerometrů.

Tabulka 1: Zvolené adresy jednotlivých akcelerometrů.

Akcelerometr	Nastavená adresa	Alternativní adresy
NXP FXOS8700CQ	`0x1D`	`0x1C 0x1E 0x1F`
NXP MMA8452Q	`0x1C`	`0x1D`
Analog Devices ADXL345	`0x53`	`0x1D`
STMicroelectronics LSM303DLH	`0x18`	`0x19`

Pro zjištění toho, jestli je akcelerometr s danou adresou připojen na sběrnici, mikrokontrolér jako Master pošle na I^2C sběrnici START sekvenci následovanou bajtem s horními sedmi bity určujícími adresu zařízení a `R/`$\overline{\texttt{W}}$ bitem 0 pro zápis. Pokud se na sběrnici nachází akcelerometr se zvolenou adresou, pak daný akcelerometr posílá ACK bit. Pokud se na sběrnici akcelerometr s danou adresou nenachází, SDA zůstane v logické 1 a situace je vyhodnocena jako NACK. Master poté posílá STOP sekvenci, a sběrnice je volná pro další komunikaci.

4.3 Zapojení akcelerometrů

Všechny vybrané akcelerometry je potřeba připojit k vývojové desce NXP FRDM-K64F minimálně pěti vodiči. Jedná se o vodiče pro SCL, SDA, signál přerušení, V_{CC} a GND.

Deska NXP FRDM-K64F s osazeným mikrokontrolérem MK64FN1M0VLL12 disponuje celkem třemi I^2C periferiemi, z nichž jen dvě jsou vyvedeny a přístupny na desce. Tato práce si však vystačí jen s jednou periferii. Na obrázku 2 vidíme, kde se na desce nachází potřebné piny pro připojení akcelerometrů k I^2C a napájení [3]. Pro I^2C komunikaci se jedná o piny `PTE24/D15` pro SCL a `PTE25/D14` pro SDA. Ke stejným pinům je připojen i akcelerometr NXP FXOS8700CQ, který je součástí desky [8].

4.3.1 Volba pinů přerušení

Při měření není možné, aby akcelerometr jako Slave posílal data na I^2C sběrnici. Je tedy potřeba, aby mikrokontrolér věděl, kdy jsou k dispozici nová data. K tomuto účelu jsou všechny vybrané akcelerometry vybaveny možností vyslání signálu přerušení. V momentě, kdy jsou k dispozici nová naměřená data, akcelerometr pošle signál mikrokontroléru, který si nová data vyžádá.

Vybrané akcelerometry mají mnoho interních funkcí, které mohou vyvolat přerušení a proto všechny disponují dvěma piny pro signalizaci nějaké události. Akcelerometr NXP FXOS8700CQ, který je součástí vývojové desky, má své dva piny přerušení `INT1` a `INT2` zapojeny do pinů mikrokontroléru `PTC6` a `PTC13`. Z toho důvodu, že k pinu `PTC6` je připojeno i tlačítko `SW2`, které se nachází na vývojové desce, se za pin zodpovědný signalizovat dostupnost nových dat zvolil pin `INT2` připojený k `PTC13`. Akcelerometr NXP MMA8452Q má piny pro přerušení na svém modulu označeny jako `I1` a `I2`. V případě akcelerometrů Analog Devices ADXL345 a STMicroelectronics LSM303DLH jsou piny pro přerušení na svých modulech označeny jako `INT1` a `INT2`. U externích modulů nezáleželo, jaký pin bude zodpovědný za signalizaci dostupnosti nových dat, a tak jim

byl vždy zvolen první pin, tedy `I1` nebo `INT1`. Tento signál přerušení pro všechny externí moduly byl připojen k desce do pinu `PTC12/D8`.

4.3.2 Volba adresy akcelerometru NXP FXOS8700CQ

Adresa akcelerometru NXP FXOS8700CQ, který je pevnou součástí desky NXP FRDM-K64F, je považována za fixní, neboť změna této adresy by vyžadovala modifikaci vývojové desky. Z technické dokumentace pro NXP FXOS8700CQ se lze dočíst, že piny `SA0` a `SA1` slouží pro změnu adresy akcelerometru [4]. Ze schématu vývojové desky lze zjistit, že pin `SA0` je připojen pull-up rezistorem k V_{CC} a pin `SA1` je připojen pull-down rezistorem ke GND [8]. Při opětovném nahlédnutí do technické dokumentace lze zjistit, že toto zapojení odpovídá adrese `0x1D`.

4.3.3 Volba adresy akcelerometru NXP MMA8452Q

Akcelerometr NXP MMA8452Q na modulu SPARKFUN ELECTRONICS INC. BOB-13926 disponuje na spodní straně dvěma kontakty, označenými jako `SA0`. Z technické dokumentace pro NXP MMA8452Q vyplývá, že pin `SA0` slouží pro změnu posledního bitu adresy. Pokud je pin `SA0` přiveden k V_{CC}, pak je výsledná adresa `0x1D`, pokud je ale přiveden ke GND, pak je adresa `0x1C` [5]. Ze schématu pro tento modul lze vyčíst, že pin `SA0` je připojen pull-up rezistorem k V_{CC}, a že oné dva kontakty slouží pro připojení pinu `SA0` ke GND [9]. Tyto dva kontakty tedy bylo potřeba spolu elektricky spojit, aby adresa nekolidovala s adresou akcelerometru NXP FXOS8700CQ.

4.3.4 Volba adresy akcelerometru Analog Devices ADXL345

Akcelerometru Analog Devices ADXL345 na modulu DFROBOT SEN0032 podporuje komunikaci jak přes I^2C, tak i SPI. Ke konfiguraci akcelerometru pro komunikaci přes I^2C je potřeba, aby pin $\overline{CS}$ byl v momentě připojení akcelerometru k napájení, také připojen k V_{CC}. Pro nastavení adresy je potřeba připojit pin `SDO/ALT ADDRESS` ke GND pro nastavení adresy akcelerometru na `0x53` [6][10]. Při připojení pinu `SDO/ALT ADDRESS` k V_{CC} by byla adresa `0x1D`, a ta by kolidovala s akcelerometrem NXP FXOS8700CQ.

4.3.5 Volba adresy akcelerometru STMicroelectronics LSM303DLH

U akcelerometru STMicroelectronics LSM303DLH na modulu DFROBOT SEN0079 nebylo třeba žádným způsobem řešit změnu adresy, neboť žádná z jeho adres nekoliduje s jiným akcelerometrem. Při pohledu do technické dokumentace lze zjistit, že adresu lze změnit pinem `SA0_A` [7]. Při pohledu do schéma zapojení jde vidět, že tento pin je pull-down rezistorem připojen ke GND [11]. Toto zapojení odpovídá adrese `0x18`. Pokud by bylo potřeba adresu změnit, tak pin `SA0_A` je vyveden na modulu jako pin `SA` a připojením tohoto pinu k V_{CC} by zvolená adresa byla `0x19`.

4.4 Konfigurace akcelerometrů

Pro porovnání vybraných akcelerometrů bylo nutné je nastavit tak, aby data z nich získaná reprezentovala to samé. Toho bylo docíleno nastavením dvou různých parametrů. Tím prvním je rozsah, v jakém budou akcelerometry data měřit. Experimenty bylo zjištěno, že rozsah $\pm 2g$ je plně dostačující pro rozpoznání chůze. Druhým parametrem je rychlost s jakou akcelerometry posílají data mikrokontroléru. Nemusí se však jednat o vzorkovací frekvenci, protože akcelerometry NXP FXOS8700CQ a NXP MMA8452Q umožňují vzorkovat data s vyšší rychlostí, než je rychlost odesílání dat, pro redukci šumu. Data jsou posílána z akcelerometrů rychlosti 50 vzorků za sekundu. Tato rychlost je nejnižší společná rychlost odesílání dat mezi vybranými akcelerometry.

4.4.1 Konfigurace akcelerometru STMicroelectronics LSM303DLH

K nastavení akcelerometru STMicroelectronics LSM303DLH byly nastaveny následující 3 registry: `CTRL_REG1_A`, `CTRL_REG3_A`, `CTRL_REG4_A`. Adresy a nastavení těchto registrů jsou v tabulce 2. Jako příklad inicializace je ukázka kódu ve výpisu 2.

Tabulka 2: Nastavení registrů akcelerometru STMicroelectronics LSM303DLH

Registr	Adresa	Vybrané nastavení
`CTRL_REG1_A`	`0x20`	`0b00100111`
`CTRL_REG3_A`	`0x22`	`0b11000010`
`CTRL_REG4_A`	`0x23`	`0b00000000`

```
status_t LSM_Init(I2C_Type *base, uint8_t slaveAddress) {
   status_t result = kStatus_Success;
   uint8_t data;

   data = 0b00000111; // power down / standby
   if((result = BOARD_I2C_Send(base, slaveAddress, LSM_CTRL_REG1_A, 1, &data,
       1)) != kStatus_Success)
      return result;

   data = 0b11000010; // active low / open drain / Data ready interrupt on INT1
   if((result = BOARD_I2C_Send(base, slaveAddress, LSM_CTRL_REG3_A, 1, &data,
       1)) != kStatus_Success)
      return result;

   data = 0b00000000; // little-endian / 2g
```

```
    if((result = BOARD_I2C_Send(base, slaveAddress, LSM_CTRL_REG4_A, 1, &data,
        1)) != kStatus_Success)
      return result;

    return result;
}
```

Výpis 2: Inicializace akcelerometru STMicroelectronics LSM303DLH

Registr `CTRL_REG1_A` slouží pro přepínání akcelerometru mezi aktivním a pohotovostním režimem. Při jakémkoliv nastavování je nejprve nutné akcelerometr přepnout do pohotovostního režimu. Jakmile je vše nastaveno, akcelerometr je možné přepnout do aktivního režimu. Dále tento registr slouží pro nastavení rychlosti odesílání dat.

Registr `CTRL_REG3_A` slouží pro elektrické nastavení vlastností signálu přerušení. Při vyvolání přerušení akcelerometr nastaví pin a signál na něm k logické 0. Dále slouží pro povolení vyslání signálu přerušení při dostupnosti nových naměřených dat a pro nastavení pinu, na který se signál přerušení pošle. V případě tohoto akcelerometru se jedná o pin `INT1`.

Registr `CTRL_REG4_A` slouží pro nastavení měřeného rozsahu zrychlení a pro nastavení pořadí bajtů výstupních dat. Zvolen byl little-endian.

4.4.2 Konfigurace akcelerometru NXP FXOS8700CQ

K nastavení akcelerometru NXP FXOS8700CQ bylo nastaveno celkem 7 registrů: `CTRL_REG1`, `CTRL_REG2`, `CTRL_REG3`, `CTRL_REG4`, `CTRL_REG5`, `XYZ_DATA_CFG`, `M_CTRL_REG1`. Adresy a nastavení těchto registrů jsou v tabulce 3.

Tabulka 3: Nastavení registrů akcelerometru NXP FXOS8700CQ

Registr	Adresa	Vybrané nastavení
CTRL_REG1	0x2A	0b00100101
CTRL_REG2	0x2B	0b00000010
CTRL_REG3	0x2C	0b00000001
CTRL_REG4	0x2D	0b00000001
CTRL_REG5	0x2E	0b00000000
XYZ_DATA_CFG	0x0E	0b00000000
M_CTRL_REG1	0x5B	0b00000000

Registr `CTRL_REG1` slouží pro nastavení rychlosti odesílání dat, nastavení módu s omezeným šumem a přepínání mezi aktivním a pohotovostním režimem. Při jakémkoliv nastavování je nejprve nutné akcelerometr přepnout do pohotovostního režimu. Jakmile je vše nastaveno, akcelerometr je možné přepnout do aktivního režimu.

Registr `CTRL_REG2` slouží pro nastavení převzorkování, kdy je vzorkovací frekvence vyšší než je výstupní frekvence dat, pro další redukci šumu.

Registr CTRL_REG3 slouží pro elektrické nastavení vlastností signálu přerušení. Při vyvolání přerušení akcelerometr nastaví pin a signál na něm k logické 0.

Registr CTRL_REG4 slouží pro povolení vyslání signálu přerušení při dostupnosti nových naměřených dat.

Registr CTRL_REG5 slouží pro nastavení pinu, na který se signál přerušení pošle. V případě tohoto akcelerometru se jedná o pin INT2.

Registr XYZ_DATA_CFG slouží pro nastavení měřeného rozsahu zrychlení.

Registr M_CTRL_REG1 je prvním kontrolním registrem pro nastavení magnetometru. Nastavením tohoto registru lze funkci magnetometru vypnout.

4.4.3 Konfigurace akcelerometru NXP MMA8452Q

Nastavení akcelerometrů NXP MMA8452Q je velice podobné nastavení akcelerometru NXP FXOS8700CQ v sekci 4.4.2. Tento akcelerometr totiž sdílí adresy registrů a význam jednotlivých bitů v daných registrech s akcelerometrem NXP FXOS8700CQ. Hlavním rozdílem je to, že tento akcelerometr, respektive toto pouzdro, v sobě neobsahuje magnetometr, a tudíž postrádá registr M_CTRL_REG1. Druhým rozdílem je to, že signál přerušení pro nová data v registru CTRL_REG5, je nastaven na pin INT1. Adresy a nastavení registrů tohoto akcelerometru jsou v tabulce 4.

Tabulka 4: Nastavení registrů akcelerometru NXP MMA8452Q

Registr	Adresa	Vybrané nastavení
CTRL_REG1	0x2A	0b00100101
CTRL_REG2	0x2B	0b00000010
CTRL_REG3	0x2C	0b00000001
CTRL_REG4	0x2D	0b00000001
CTRL_REG5	0x2E	0b00000001
XYZ_DATA_CFG	0x0E	0b00000000

4.4.4 Konfigurace akcelerometru Analog Devices ADXL345

Pro nastavení akcelerometru Analog Devices ADXL345 bylo nastaveno 5 registrů: BW_RATE, POWER_CTL, INT_ENABLE, INT_MAP, DATA_FORMAT. Adresy a nastavení těchto registrů jsou v tabulce 5.

Registr BW_RATE slouží pro nastavení rychlosti odesílání dat.

Registr POWER_CTL slouží pro přepínání akcelerometru mezi aktivním a pohotovostním režimem. Při jakémkoliv nastavování je nejprve nutné akcelerometr přepnout do pohotovostního režimu. Jakmile je vše nastaveno, akcelerometr je možné přepnout do aktivního režimu.

Registr INT_ENABLE slouží pro povolení vyslání signálu přerušení při dostupnosti nových naměřených dat.

Tabulka 5: Nastavení registrů akcelerometru Analog Devices ADXL345

Registr	Adresa	Vybrané nastavení
BW_RATE	0x2C	0b00001001
POWER_CTL	0x2D	0b00001000
INT_ENABLE	0x2E	0b10000000
INT_MAP	0x2F	0b00000000
DATA_FORMAT	0x31	0b00101100

Registr INT_MAP slouží pro nastavení pinu, na který se signál přerušení pošle. V případě tohoto akcelerometru se jedná o pin INT1.

Registr DATA_FORMAT slouží pro elektrické nastavení vlastností signálu přerušení. Při vyvolání přerušení akcelerometr nastaví pin a signál na něm k logické 0. Dále tento registr slouží pro nastavení měřeného rozsahu zrychlení.

4.5 Formát přenášených dat

Všechny vybrané akcelerometry, bez ohledu na rozlišení, ukládají naměřené hodnoty do šesti registrů. V těchto šesti registrech je uložena informace o velikosti naměřeného zrychlení pro osy x, y a z. Každé ose jsou věnovány dva bajty. Tyto 16bitové hodnoty jsou reprezentovány ve dvojkovém doplňku a jsou zarovnány doleva.

Všechny akcelerometry umožňují čtení všech těchto šesti registrů najednou. Pro získání naměřených dat tedy stačí začít čtení na první adrese registrů naměřených dat. V případě akcelerometrů NXP FXOS8700CQ a NXP MMA8452Q se jedná o adresu 0x01. Pro akcelerometry Analog Devices ADXL345 se jedná o adresu 0x32 a pro STMicroelectronics LSM303DLH se jedná o adresu 0x28. V případě akcelerometru STMicroelectronics LSM303DLH je navíc potřeba pro čtení více bajtů nastavit nejvyšší bit čteného registru jako 1, tedy přidat k adrese registru hodnotu 0x80. Výsledná adresa pro čtení naměřených dat je tedy 0xA8.

Akcelerometry NXP FXOS8700CQ a NXP MMA8452Q posílají data ve formátu big-endian. To znamená, že pro každou osu je nejprve poslán MSB a následně LSB. Akcelerometry Analog Devices ADXL345 a STMicroelectronics LSM303DLH posílají data ve formátu little-endian, kde je pořadí posílaných bajtů opačné.

5 Ukládání dat

V momentě kdy má akcelerometr nová data k dispozici, tak vyšle signál přerušení mikrokontroléru, který si data po I^2C sběrnici vyzvedne.

Pro příjem přerušení bylo povoleno přerušení na sestupné hraně na GPIO pinech `PTC12/D8` a `PTC13`. Oba tyto piny jsou připojeny k portu C, a tak bylo také potřeba povolit přerušení generováno tímto portem. Toto se provádí povolením přerušení v NVIC pro port C.

Jakmile je přijat signál přerušení, činnost procesoru je pozastavena a je vykonána funkce spojena s obsluhou přerušení. Tato funkce zahájí po I^2C sběrnici čtení šesti registrů pro získání velikosti naměřeného zrychlení ve všech třech osách. Tyto data jsou ukládaná do šestibajtového bufferu. Následně činnost procesoru pokračuje na místě, kde bylo přerušení vyvoláno. Implementace obsluhy přerušení je ve výpisu 3. Jakmile je přenos šesti bajtů po I^2C sběrnici dokončen, jsou tyto bajty umístěny do hlavního kruhového bufferu ve správném pořadí, viz výpis 4. Mikrokontrolér na desce NXP FRDM-K64F ukládá data ve své paměti ve formě little-endian.

```
void PORTC_IRQHandler(void) {
   masterXfer.flags = kI2C_TransferDefaultFlag;
   masterXfer.slaveAddress = accel_address;
   masterXfer.direction = kI2C_Read;
   masterXfer.subaddressSize = 1;
   masterXfer.data = accel_data;
   masterXfer.dataSize = sizeof(accel_data);

   switch(accel_address) {
   case ACCEL_ADXL_ADDRESS:
      masterXfer.subaddress = 0x32U;
      break;
   case ACCEL_MMA_ADDRESS:
      masterXfer.subaddress = 0x01U;
      break;
   case ACCEL_LSM_ADDRESS:
      masterXfer.subaddress = 0x28U | 0x80U;
      break;
   case ACCEL_FXOS_ADDRESS:
      masterXfer.subaddress = 0x01U;
      break;
   }

   I2C_MasterTransferNonBlocking(I2C0, &masterHandle, &masterXfer);
```

```
   i2c_non_blocking_running = true;

   PORT_ClearPinsInterruptFlags(PORTC, PORT_GetPinsInterruptFlags(PORTC));

#if defined __CORTEX_M && (__CORTEX_M == 4U)
    __DSB();
#endif
}
```

Výpis 3: Obsluha přerušení pro zahájení vyzvednutí dat

```
static void i2c_master_callback(I2C_Type *base, i2c_master_handle_t *handle,
    status_t status, void *userData)
{
    if (status == kStatus_Success)
    {
      switch(accel_address) {
      case ACCEL_FXOS_ADDRESS: case ACCEL_MMA_ADDRESS: // big endian
         buffer[buffer_index++] = (int16_t)(((uint16_t)accel_data[0] << 8) | (
             uint16_t)accel_data[1]);
         buffer_index &= BUFFER_SIZE-1U;

         buffer[buffer_index++] = (int16_t)(((uint16_t)accel_data[2] << 8) | (
             uint16_t)accel_data[3]);
         buffer_index &= BUFFER_SIZE-1U;

         buffer[buffer_index++] = (int16_t)(((uint16_t)accel_data[4] << 8) | (
             uint16_t)accel_data[5]);
         buffer_index &= BUFFER_SIZE-1U;
         break;
      default: // little endian
         buffer[buffer_index++] = *((int16_t*)accel_data);
         buffer_index &= BUFFER_SIZE-1U;

         buffer[buffer_index++] = *((int16_t*)accel_data+1);
         buffer_index &= BUFFER_SIZE-1U;

         buffer[buffer_index++] = *((int16_t*)accel_data+2);
         buffer_index &= BUFFER_SIZE-1U;
      }
```

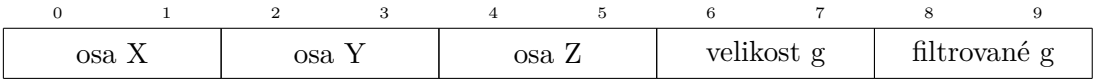
```
      buffer_index += ADDITIONAL_CHANNELS;
      buffer_index &= BUFFER_SIZE-1U;
    }
    else {
      error = __LINE__;
    }
   i2c_non_blocking_running = false;
}
```

Výpis 4: Uložení dat do kruhového bufferu

Hlavním bufferem pro zpracování a ukládání dat je kruhový buffer. Jedná se o buffer typu fronta, kdy data v něm jsou zpracována v pořadí, v jakém byly do bufferu vloženy. Data ukládaná do tohoto bufferu jsou datového typu `int16_t`. Jeden naměřený vzorek se celkem skládá z pěti kanálů. Každému tomuto kanálu jsou věnovány dva bajty. V prvních třech kanálech jsou uloženy nezpracované hodnoty os x, y a z. Ve čtvrtém kanále je uložená vypočtená hodnota celkového tíhového zrychlení g působící na akcelerometr. Ta se vypočítá aplikováním následujícího vzorce:

$$a = \sqrt{x^2 + y^2 + z^2} \tag{1}$$

Poslední pátý kanál slouží pro ukládání filtrovaných dat, získaných filtrováním předchozího kanálu, viz sekce 8. Data z tohoto kanálu jsou následně používána k detekci a počítání kroků, viz sekce 9. Obrázek 5 zobrazuje strukturu jednoho vzorku ukládaných dat.



Obrázek 5: Struktura ukládaných dat

Výpočet velikosti tíhového zrychlení a filtrování signálu není součástí obsluhy přerušení, ale hlavní smyčky programu, viz výpis 5. Nad kruhovým bufferem pracuje několik indexů. Index `buffer_index` se posouvá vždy s nově přijatými daty o počet pozic roven počtu kanálů. Index `processed_index` ukazuje na pozici již vypočtených a filtrovaných dat. V momentě kdy přijdou nová data, index `processed_index` se posouvá tak dlouho, dokud není roven `buffer_index`.

```
while(processed_index != buffer_index) {
   switch(current_channel % TOTAL_CHANNELS) {
   case 0:
      fx = buffer[processed_index++];
      processed_index &= BUFFER_SIZE-1U;
      break;
   case 1:
```

```
        fy = buffer[processed_index++];
        processed_index &= BUFFER_SIZE-1U;
        break;
    case 2:
        fz = buffer[processed_index++];
        processed_index &= BUFFER_SIZE-1U;
        break;
    case 3:
        buffer[processed_index++] = sqrtf(fx*fx + fy*fy + fz*fz) + INT16_MIN;
        processed_index &= BUFFER_SIZE-1U;
        break;
    case 4:
        buffer[processed_index++] = filtered_signal_value;
        processed_index &= BUFFER_SIZE-1U;
        break;
    default:
        buffer[processed_index++] = 0;
        processed_index &= BUFFER_SIZE-1U;
        break;
    }
    if(++current_channel >= TOTAL_CHANNELS) {
        current_channel = 0;
    }
}
```

Výpis 5: Výpočet hodnot zbývajících kanálů

Struktura uložených dat je stejná jakou používá WAV audio formát. Tento formát je primárně určen pro bezeztrátové ukládání zvuku, avšak díky jednoduchosti implementace a pružnosti tohoto formátu, je tento formát použit i v této práci pro ukládání naměřených dat. Výhodou tohoto přístupu je, že k vizualizaci naměřených dat z akcelerometrů lze využít běžný audio editor. V této práci byl použit multiplatformní open source editor digitálního zvuku Audacity ve verzi 2.3.0.

Velikost WAV hlavičky je 44 bajtů [12]. Jsou v ní uloženy informace o velikosti souboru, počtu kanálů, vzorkovací frekvenci a velikosti jednoho vzorku. Za touto hlavičkou pak následují samotná data. Jelikož je třeba znát velikost souboru v momentě vytvoření této hlavičky, tak se ukládá až v momentě, kdy je měření ukončeno. Proměnná `Subchunk2Size` slouží pro určení velikosti datové části v bajtech. Vytvoření hlavičky je ve výpisu 6.

```
void Create_Wave_Header(void *buff, uint16_t NumChannels, uint32_t SampleRate,
    uint16_t BitsPerSample, uint32_t Subchunk2Size) {
    memcpy(buff, "RIFF----WAVEfmt ----------------data----", 44);
    *((uint32_t*)(buff+4)) = 36 + Subchunk2Size; // ChunkSize
    *((uint32_t*)(buff+16)) = 16; // Subchunk1Size
    *((uint16_t*)(buff+20)) = 1; // AudioFormat
    *((uint16_t*)(buff+22)) = NumChannels; // NumChannels
    *((uint32_t*)(buff+24)) = SampleRate; // SampleRate
    *((uint32_t*)(buff+28)) = SampleRate * NumChannels * (BitsPerSample/8); //
        ByteRate
    *((uint16_t*)(buff+32)) = NumChannels * (BitsPerSample/8); // BlockAlign
    *((uint16_t*)(buff+34)) = BitsPerSample; // BitsPerSample
    *((uint32_t*)(buff+40)) = Subchunk2Size; // Subchunk2Size
}
```

Výpis 6: Inicializace WAV hlavičky

Data jsou následně ukládána na micro SD kartu připojenou k vývojové desce. Ukládání dat z kruhového bufferu probíhá vždy po polovinách jeho velikosti, viz výpis 7. Důvod je ten, že když se zaplní jedna polovina bufferu, tak při jeho zápisu na SD kartu akcelerometr nemusí čekat na uložení bufferu, neboť je stále kam ukládat data.

Pro zápis dat na micro SD kartu byl použit softwarový modul FatFs [14]. Tento modul umožňuje práci se souborovými systémy FAT a exFAT. Modul je kompletně oddělen od fyzických zařízení a díky tomu je nezávislý na platformě a použitém médiu ukládání dat. Kvůli tomu, že nízkoúrovňová komunikace není součástí FatFs modulu, je na výrobcích (v tomto případě NXP) aby implementovali funkce pro nízkoúrovňový přístup.

Jména ukládaných WAV souborů jsou generovány podle počtu uložených souborů a podle připojeného akcelerometru, kde první čtyři znaky jsou rezervovány pro číslo měření a další čtyři znaky jsou pro označení připojeného akcelerometru, např. `0000fxos.wav`, `0001-lsm.wav` a `0002adxl.wav`. Společně s těmito soubory jsou také vytvářený TXT soubory se stejným názvem, které v sobě obsahují počet naměřených kroků vývojovou deskou.

```
if(previous_state != (processed_index >= BUFFER_SIZE/2)) {
   LED_BLUE_ON();
   previous_state = (processed_index >= BUFFER_SIZE/2); // previous_state is
       now current state
   if(f_write(&file_bin, buffer+((previous_state)?(0):(BUFFER_SIZE/2)), (sizeof
       (buffer))/2, &bytes_written) != FR_OK) {
      error = __LINE__;
```

```
            LED_BLUE_OFF();
            break; // breaks the main loop
        }
        total_file_size += bytes_written;
        LED_BLUE_OFF();
    }
```

Výpis 7: Ukládání dat na SD kartu

Jeden z možných způsobů, jak získat naměřená data z SD karty, je jejím vyjmutím z vývojové desky a vložením do čtečky paměťových karet. Druhý způsob, jak se dostat k naměřeným datům, je připojením samotné vývojové desky k počítači pomocí USB kabelu. Je nutné, aby kabel byl zapojen do micro USB konektoru, který je přímo spojen s mikrokontrolérem. Po připojení vývojové desky k počítači je deska rozpoznána jako USB mass storage device, a chová se jako standardní USB flash disk. K implementaci této funkce byl použit projekt `dev_msc_sdcard_bm`, který je dostupný společně s SDK pro vývojovou desku NXP FRDM-K64F. Toto řešení bylo převzato pouze s minimálními úpravami.

6 Měření testovacích dat

Pro získání dat chůze proběhlo několik různých měření se všemi čtyřmi akcelerometry. Data byla zaznamenána na celkem čtyřech různých místech a byla zaznamenána jak při držení vývojové desky s akcelerometry v ruce, tak při umístění v kapse.

Každé měření bylo doprovázeno dvěma krokoměry. Výstupem těchto krokoměrů byl počet kroků, které později sloužily pro porovnání počtu kroků při implementaci filtrů a vlastního řešení detekce kroků, viz sekce 9. Prvním z krokoměrů byl krokoměr Xiaomi Mi Band. Jedná se o krokoměr ve formě náramku, tudíž veškerá měření s ním provedena byla pořízená s tímto náramkem na zápěstí. Druhým použitým krokoměrem byla mobilní aplikace Pedometer - Step Counter by Simple Design Ltd. provozována na mobilním telefonu Nokia 6.1 DualSIM (TA-1043). Toto zařízení bylo vždy umístěno na společném místě s vývojovou deskou v kapse nebo v ruce.

Z výsledků naměřených v této sekci je patrné, že výsledky podávané jednotlivými akcelerometry v jednotlivých situacích si jsou velice podobné. Rozdíly mezi jednotlivými akcelerometry jsou způsobeny měřením akcelerometrů v různých časech, a nikoliv současně. Všechny vybrané akcelerometry tedy byly shledány jako vhodné pro použití v krokoměru.

Data zobrazená v této sekci zobrazují celkovou velikost tíhového zrychlení g působící na akcelerometr. Obrázky samotné byly pořízeny nahráním naměřených dat do programu Audacity.

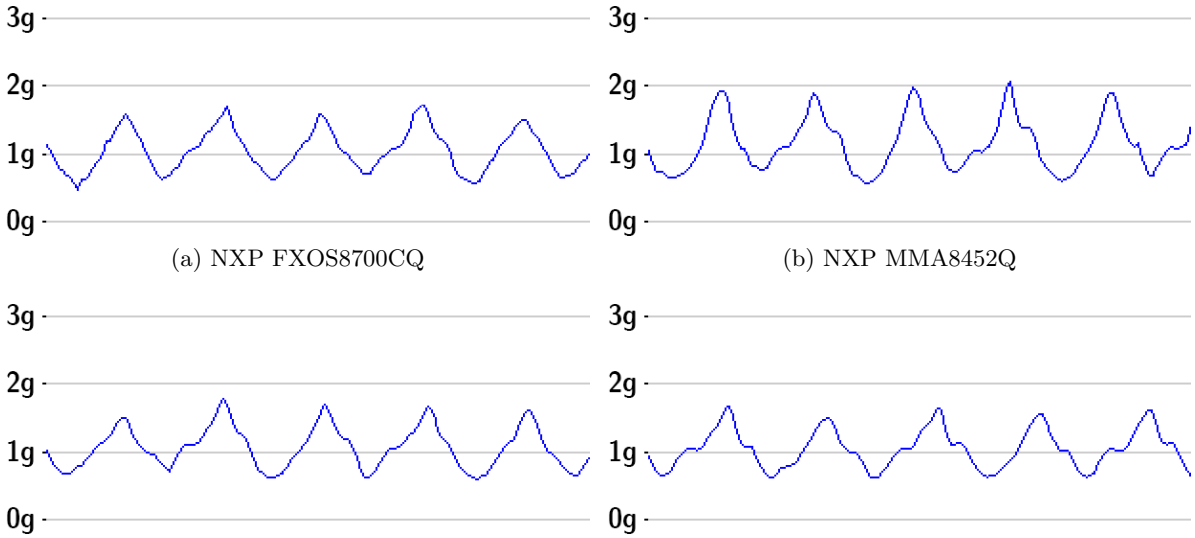
6.1 Chůze po rovině

Chůze po rovině je ideálním případem, na kterém se chůze může vyskytnou. Měření proběhlo na přibližně jeden kilometr dlouhém vyasfaltovaném okruhu a každé z nich trvalo přibližně 11 minut. Při měření chůze se nebylo třeba ničemu vyhýbat a chůze samotná měla velmi stabilní tempo. Obrázek 6 zobrazuje okruh, na kterém se měření provádělo. Přibližné souřadnice zvoleného okruhu jsou 49.6712N, 18.3725E.

Obrázky 7 a 8 zobrazují krátké, třísekundové úseky naměřené chůze na rovině pro každý z vybraných akcelerometrů. Obrázek 7 zobrazuje signál při držení vývojové desky v ruce. Při pohledu na naměřený signál lze jednoduše rozpoznat jednotlivé kroky chůze. Obrázek 8 zobrazuje signál při nošení vývojové desky v kapse. Na rozdíl od předchozího obrázku se jednotlivé kroky už nedají tak snadno rozpoznat. Na signálu jdou vidět prudké změny amplitudy v krátkém čase. Díky tomu, že všechny obrázky zobrazují tři sekundy naměřeného signálu, během kterých nastane přibližně pět kroků, lze jednotlivé kroky identifikovat. Tyto prudké změny jsou s nejvyšší pravděpodobností způsobeny volným pohybem a skákáním vývojové desky a připojených akcelerometrů v kapse při dopadu nohy na zem. Při porovnání signálu jednotlivých akcelerometrů mezi sebou lze říct, že signály si jsou tvarem a amplitudou velice podobné.

Tabulky 6 a 7 zobrazují pro jednotlivá měření délku měřené chůze a počet kroků naměřených dostupnými krokoměry. Ve všech případech je rozdíl v počtu kroků při jednotlivých měřeních mezi jednotlivými krokoměry menší než 2%.

Obrázek 6: Úsek, na kterém probíhala chůze na rovině



(a) NXP FXOS8700CQ

(b) NXP MMA8452Q

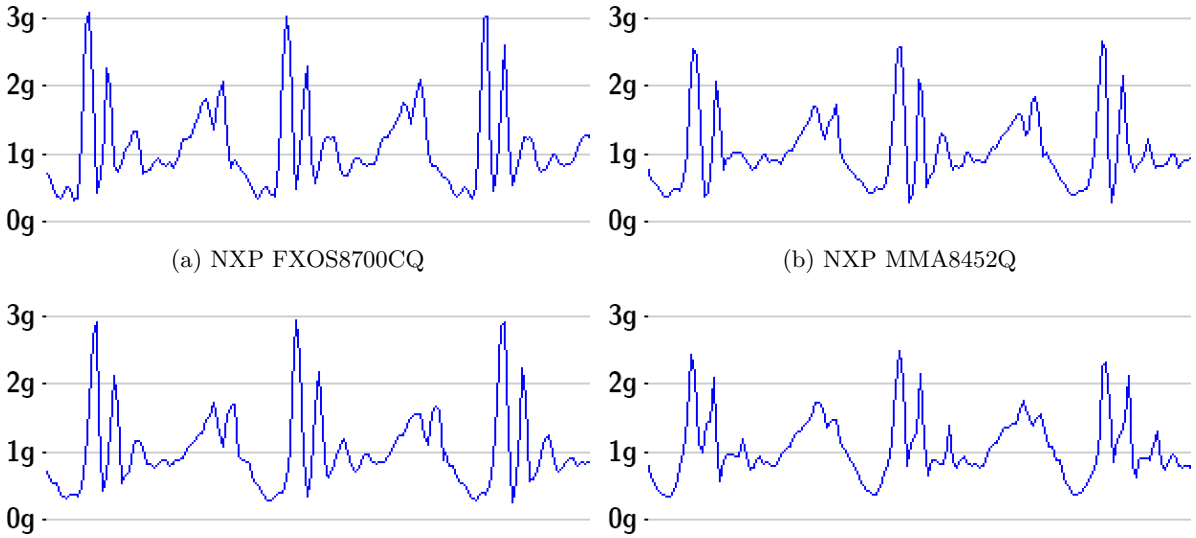
(c) Analog Devices ADXL345

(d) STMicroelectronics LSM303DLH

Obrázek 7: Třísekundové ukázky chůze po rovině s krokoměrem v ruce

Tabulka 6: Počet kroků při chůzi po rovině s krokoměrem v ruce

Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	10m 41s	1165	1176
NXP MMA8452Q	10m 44s	1187	1185
Analog Devices ADXL345	10m 56s	1200	1185
STMicroelectronics LSM303DLH	11m 19s	1183	1183



(a) NXP FXOS8700CQ

(b) NXP MMA8452Q

(c) Analog Devices ADXL345

(d) STMicroelectronics LSM303DLH

Obrázek 8: Třísekundové ukázky chůze po rovině s krokoměrem v kapse

Tabulka 7: Počet kroků při chůzi po rovině s krokoměrem v kapse

Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	10m 43s	1168	1191
NXP MMA8452Q	10m 55s	1192	1192
Analog Devices ADXL345	10m 54s	1191	1182
STMicroelectronics LSM303DLH	11m 26s	1201	1201

6.2 Chůze po trávě

Měření proběhlo na přibližně 580 metrů dlouhém travnatém úseku vedoucí podél řeky. Tento úsek měl i místy malé malé nerovnosti. Obrázek 9 zobrazuje úsek, na kterém se měření provádělo. Přibližné souřadnice měřené trasy jsou 49.6732N, 18.3647E

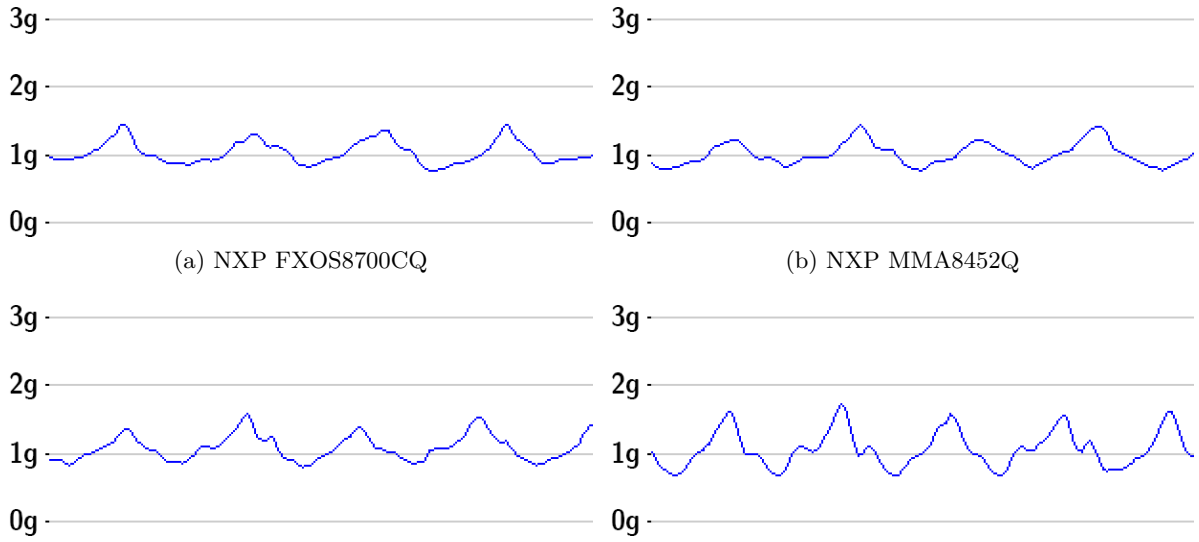
Obrázky 10 a 11 zobrazují krátké třísekundové úseky naměřené chůze na rovině pro každý z vybraných akcelerometrů. Při porovnáni signálu s chůzí po vyasfaltované rovině, viz sekce 6.1, jde vidět značný útlum naměřeného signálu, jak při držení vývojové desky a připojených akcelerometrů v ruce, tak při jejich umístění v kapse. Z naměřeného signálu při držení desky v ruce, viz obrázek 10, jsou jednotlivé kroky stále jednoduše identifikovatelné. Signál při měření s deskou umístěnou v kapse, viz obrázek 11, trpí stejným neduhem jako signál při měření na vyasfaltované rovině,viz sekce 6.1.

Tabulky 8 a 9 zobrazují pro jednotlivá měření délku měřené chůze a počet kroků naměřených dostupnými krokoměry. Rozdíly v počtu naměřených kroků při jednotlivých měřeních mezi jednotlivými krokoměry je vyšší, než v případě chůze po vyasfaltované rovině. Rozdíl procentuálně dosahuje hodnoty až 6,35% při měření chůze s akcelerometrem NXP MMA8452Q umístěným v kapse.

Obrázek 9: Úsek, na kterém probíhala chůze po trávě

Tabulka 8: Počet kroků při chůzi po trávě s krokoměrem v ruce

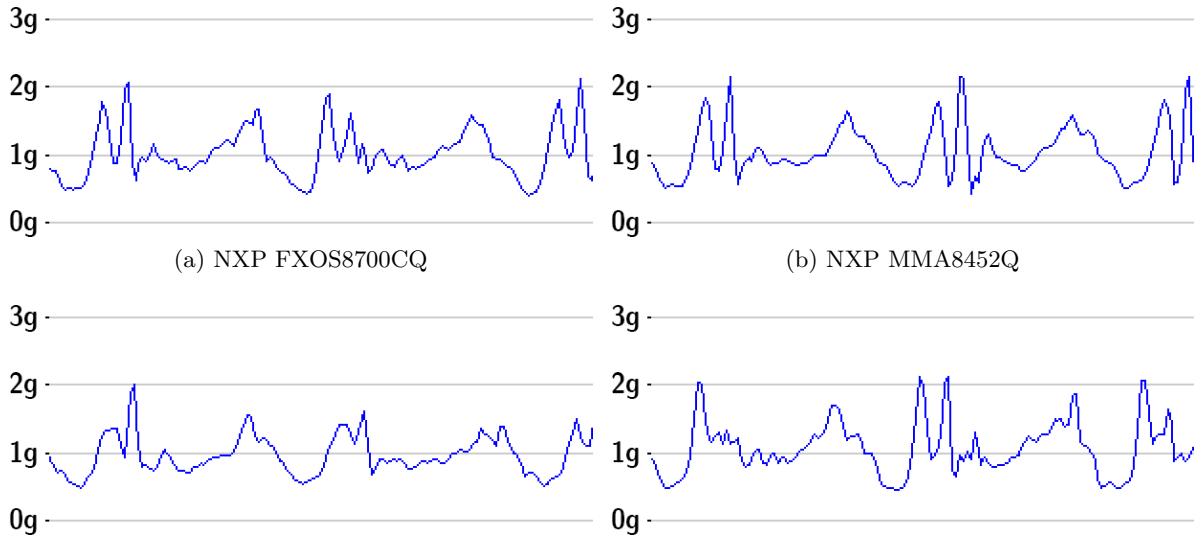
Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	8m 41s	768	787
NXP MMA8452Q	8m 34s	729	759
Analog Devices ADXL345	8m 36s	744	774
STMicroelectronics LSM303DLH	7m 28s	726	724



(a) NXP FXOS8700CQ (b) NXP MMA8452Q

(c) Analog Devices ADXL345 (d) STMicroelectronics LSM303DLH

Obrázek 10: Třísekundové ukázky chůze po trávě s krokoměrem v ruce



(a) NXP FXOS8700CQ (b) NXP MMA8452Q

(c) Analog Devices ADXL345 (d) STMicroelectronics LSM303DLH

Obrázek 11: Třísekundové ukázky chůze po trávě s krokoměrem v kapse

Tabulka 9: Počet kroků při chůzi po trávě s krokoměrem v kapse

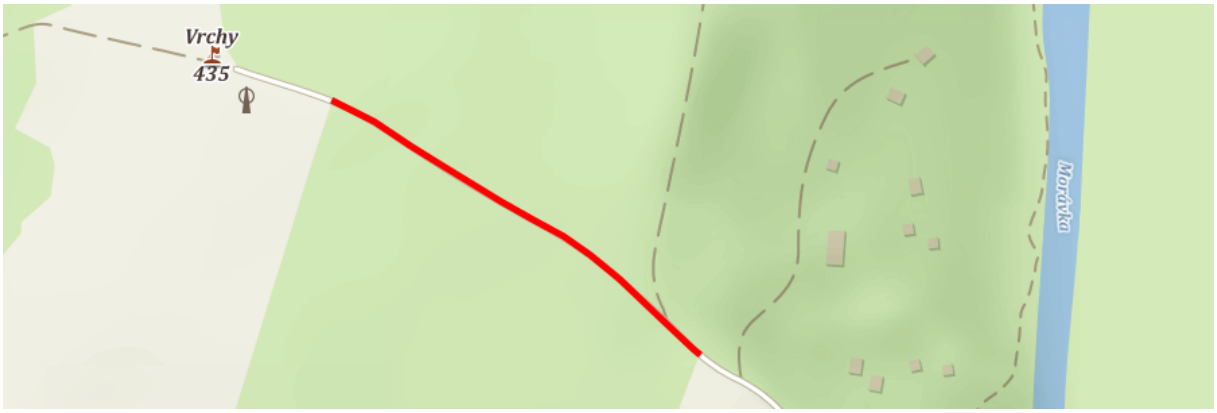
Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	8m 18s	728	763
NXP MMA8452Q	8m 18s	724	770
Analog Devices ADXL345	7m 45s	710	738
STMicroelectronics LSM303DLH	7m 14s	715	709

6.3 Chůze do a z kopce

Měření proběhlo na přibližně 260 metrů dlouhém úseku po zpevněné lesní cestě s 10% stoupáním. Obrázek 12 zobrazuje na mapě úsek, na kterém se měření provádělo. Přibližné souřadnice tohoto úseku jsou 49.6559N, 18.4169E.

Měření pro všechny akcelerometry proběhlo jak při chůzi do kopce, tak i při chůze z kopce. Obrázky 13, 14, 15 a 16 zobrazují krátké, třísekundové úseky naměřené chůze. Při držení akcelerometrů v ruce při chůzi do kopce a z kopce si jsou signály svým tvarem tvarem velmi podobné. Naměřený signál chůze do kopce je mírně slabší, než signál při chůzi z kopce. Na signálu naměřeném při umístění akcelerometrů v kapse, se na něm opět projevují velké změny v krátkém čase způsobené jeho umístěním v kapse. I v tomto případě je síla signálu při chůzi do kopce trochu nižší, než v případě chůze z kopce.

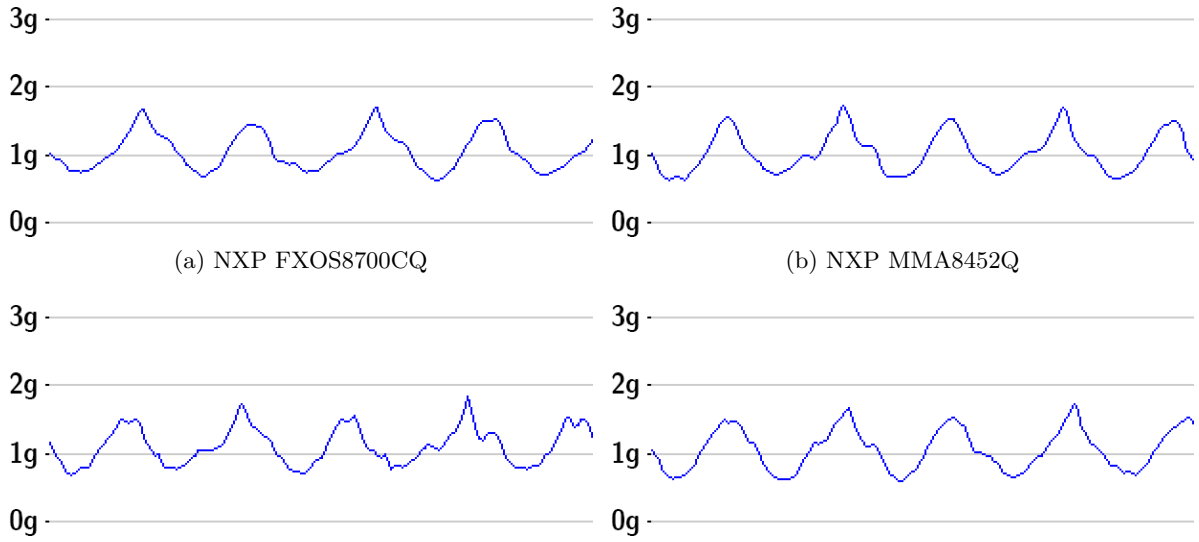
Tabulky 10, 11, 12 a 13 zobrazují počet naměřených kroků krokoměry při jednotlivých měřeních akcelerometrů. Rozdíl v počtu naměřených kroků při jednotlivých měřeních mezi jednotlivými krokoměry je menší než 2%.



Obrázek 12: Úsek, na kterém probíhala chůze do kopce a z kopce

Tabulka 10: Počet kroků při chůzi do kopce s krokoměrem v ruce

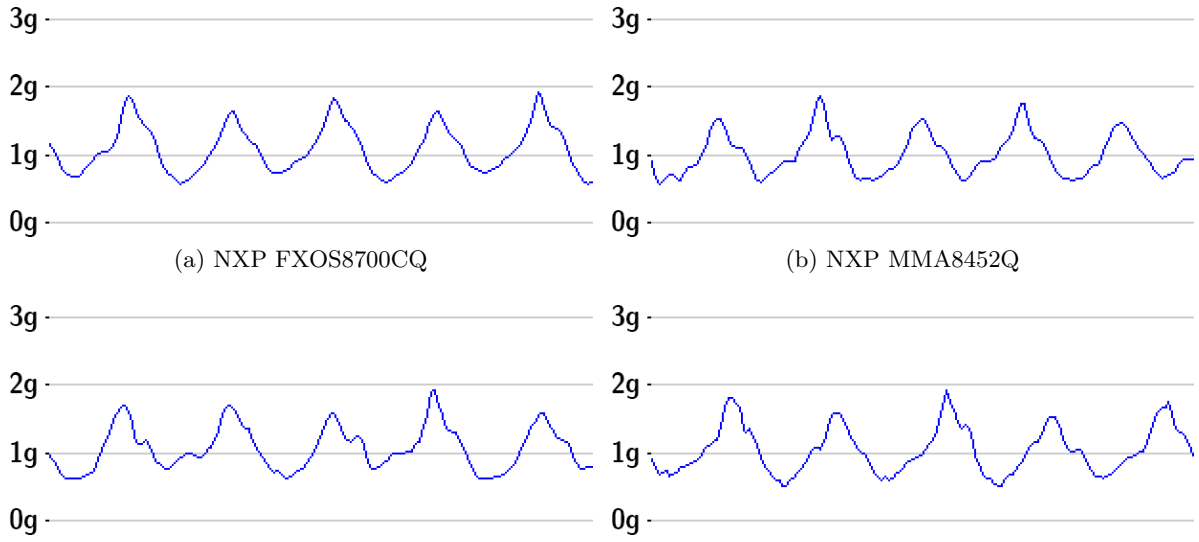
Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	3m 35s	345	351
NXP MMA8452Q	3m 25s	339	337
Analog Devices ADXL345	3m 31s	343	339
STMicroelectronics LSM303DLH	3m 30s	338	336



(a) NXP FXOS8700CQ (b) NXP MMA8452Q

(c) Analog Devices ADXL345 (d) STMicroelectronics LSM303DLH

Obrázek 13: Třísekundové ukázky chůze do kopce s krokoměrem v ruce



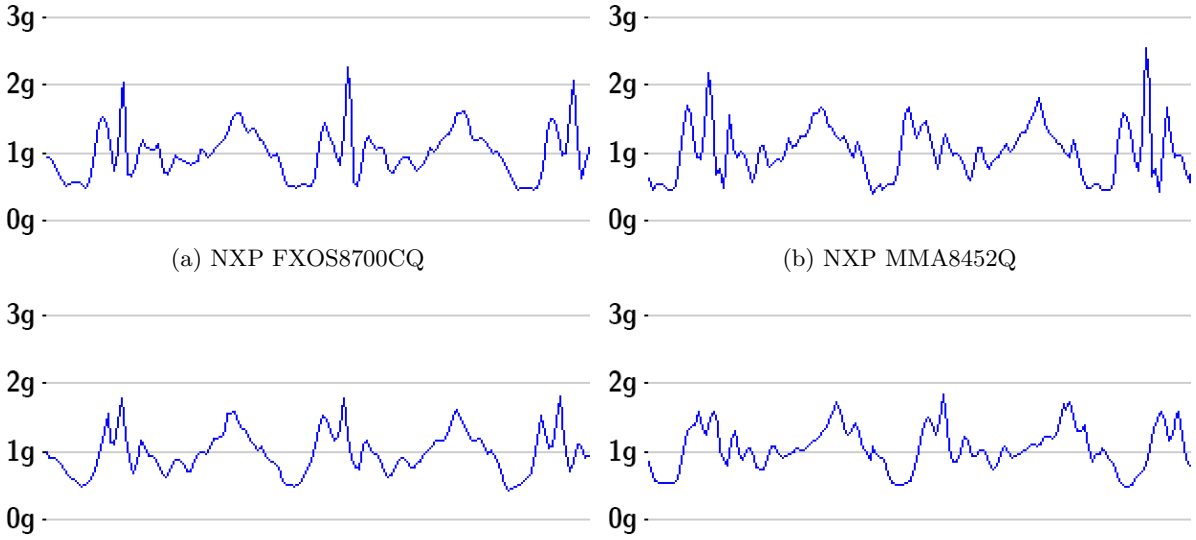
(a) NXP FXOS8700CQ (b) NXP MMA8452Q

(c) Analog Devices ADXL345 (d) STMicroelectronics LSM303DLH

Obrázek 14: Třísekundové ukázky chůze z kopce s krokoměrem v ruce

Tabulka 11: Počet kroků při chůzi z kopce s krokoměrem v ruce

Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	3m 1s	325	325
NXP MMA8452Q	2m 57s	316	315
Analog Devices ADXL345	3m 4s	319	318
STMicroelectronics LSM303DLH	3m 8s	321	318



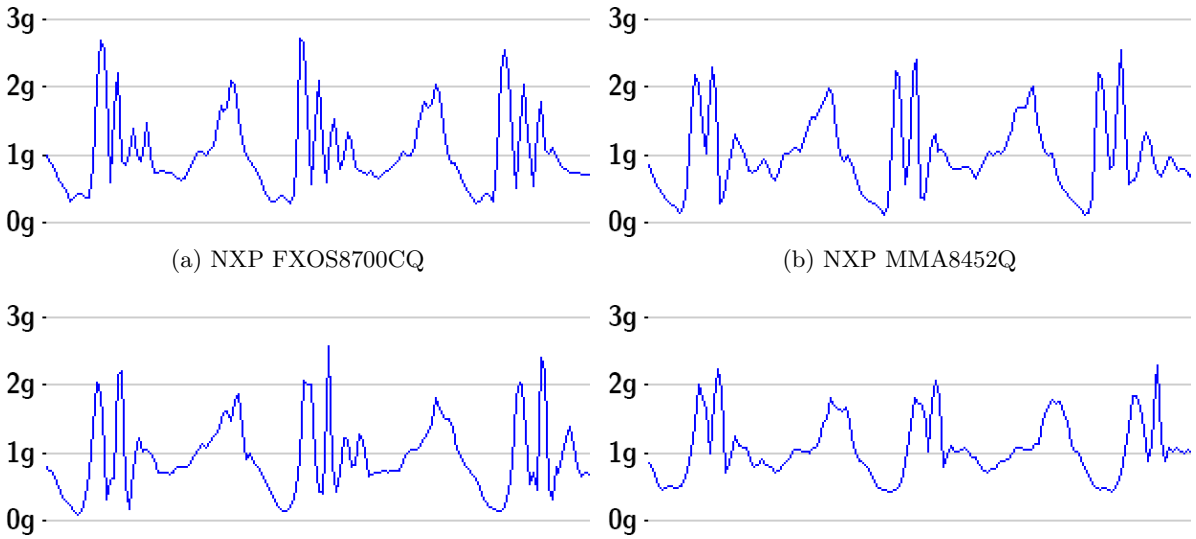
(a) NXP FXOS8700CQ

(b) NXP MMA8452Q

(c) Analog Devices ADXL345

(d) STMicroelectronics LSM303DLH

Obrázek 15: Třísekundové ukázky chůze do kopce s krokoměrem v kapse



(a) NXP FXOS8700CQ

(b) NXP MMA8452Q

(c) Analog Devices ADXL345

(d) STMicroelectronics LSM303DLH

Obrázek 16: Třísekundové ukázky chůze z kopce s krokoměrem v kapse

Tabulka 12: Počet kroků při chůzi do kopce s krokoměrem v kapse

Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	3m 40s	352	350
NXP MMA8452Q	3m 31s	347	345
Analog Devices ADXL345	3m 27s	348	347
STMicroelectronics LSM303DLH	3m 39s	347	337

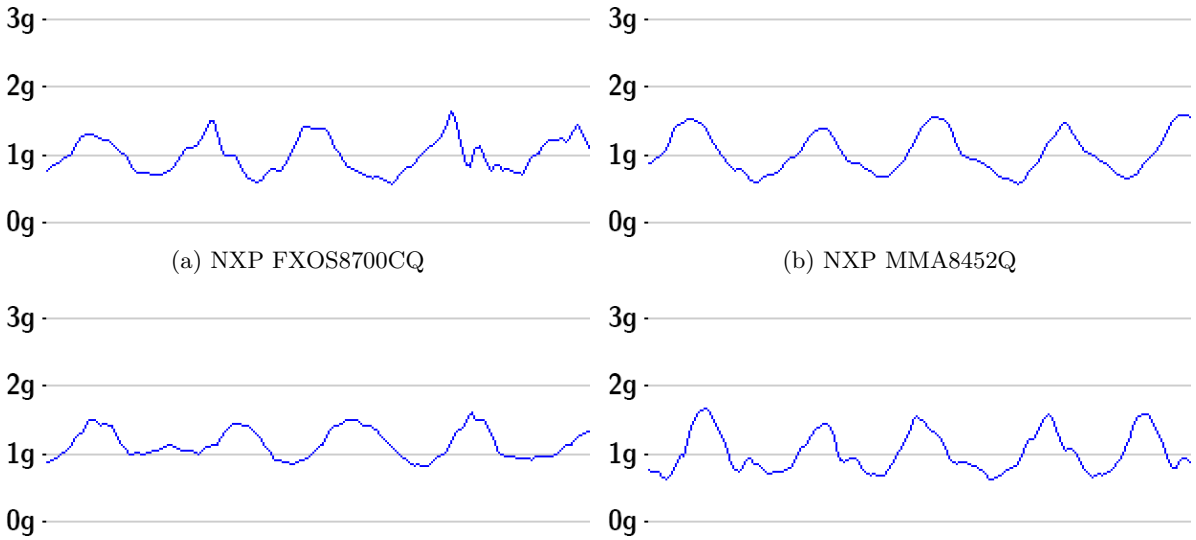
Tabulka 13: Počet kroků při chůzi z kopce s krokoměrem v kapse

Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	2m 59s	322	320
NXP MMA8452Q	3m 6s	325	322
Analog Devices ADXL345	3m 5s	326	319
STMicroelectronics LSM303DLH	3m 16s	330	326

6.4 Chůze do a ze schodů

Chůze do schodů a ze schodů byla naměřená se všemi akcelerometry na schodech panelového domu. Při každém měření bylo uděláno na schodišti přibližně 130 kroků. Obrázky 17, 18, 19 a 20 zobrazují krátké, třísekundové úseky naměřené chůze. Při držení měřených akcelerometrů v ruce jsou jednotlivé kroky chůze snadno rozpoznatelné, a to jak při chůzi do schodů, tak i při chůzi ze schodů. V případě umístění akcelerometrů v kapse, se na signálu projevuje velké množství šumu.

Tabulky 14, 15, 16 a 17 zobrazují počet naměřených kroků krokoměry při jednotlivých měřeních akcelerometrů. Při měření chůze ze schodů s akcelerometry v kapse, viz tabulka 17, bylo naraženo na limity krokoměru na mobilním telefonu. V jednom z měření bylo dokonce napočítáno jen 14 kroků, což je k očekávanému počtu 130 velice vzdáleno.



(a) NXP FXOS8700CQ

(b) NXP MMA8452Q

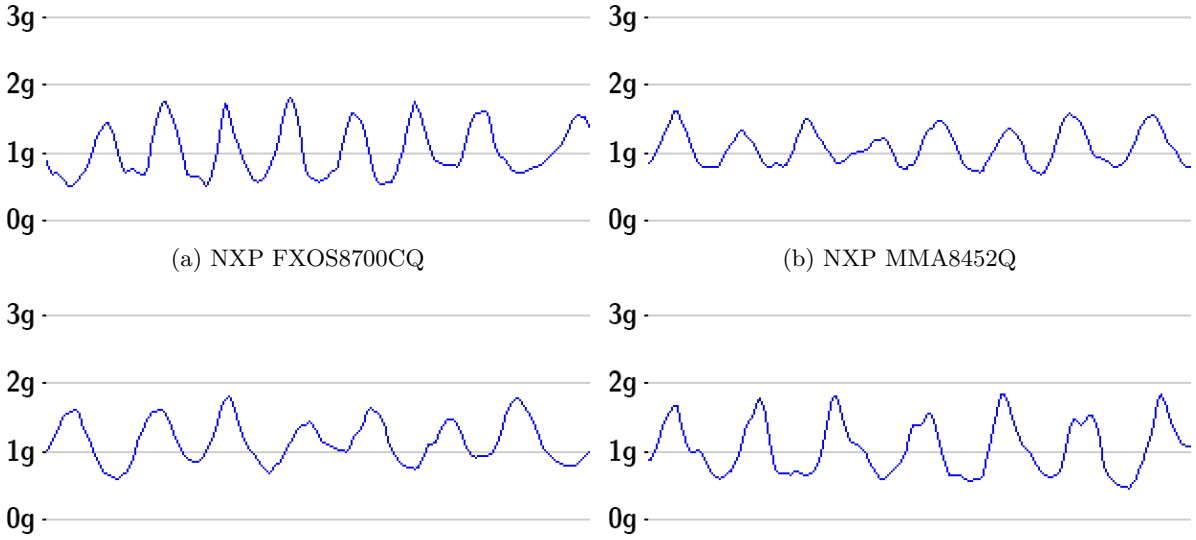
(c) Analog Devices ADXL345

(d) STMicroelectronics LSM303DLH

Obrázek 17: Třísekundové ukázky chůze do schodů s krokoměrem v ruce

Tabulka 14: Počet kroků při chůzi do schodů s krokoměrem v ruce

Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	1m 10s	112	106
NXP MMA8452Q	1m 25s	128	132
Analog Devices ADXL345	1m 28s	125	122
STMicroelectronics LSM303DLH	1m 27s	127	119



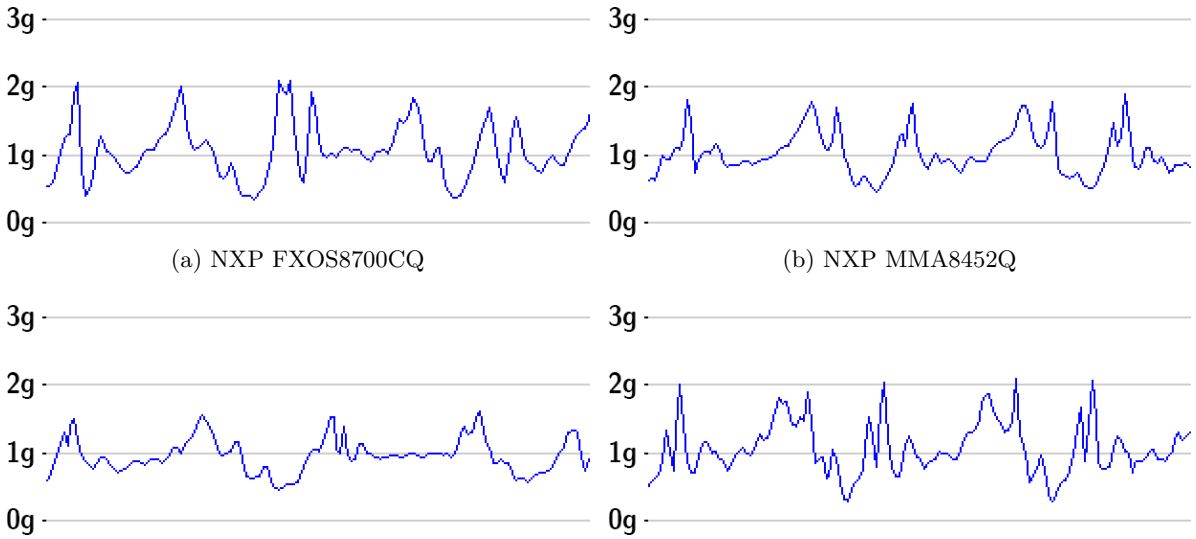
(a) NXP FXOS8700CQ

(b) NXP MMA8452Q

(c) Analog Devices ADXL345

(d) STMicroelectronics LSM303DLH

Obrázek 18: Třísekundové ukázky chůze ze schodů s krokoměrem v ruce



(a) NXP FXOS8700CQ

(b) NXP MMA8452Q

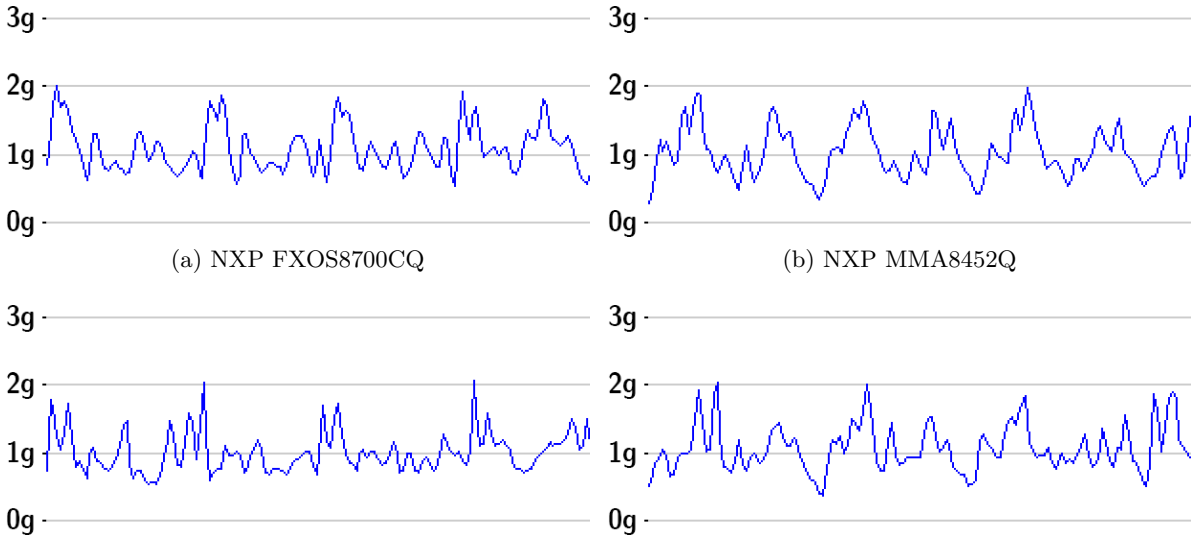
(c) Analog Devices ADXL345

(d) STMicroelectronics LSM303DLH

Obrázek 19: Třísekundové ukázky chůze do schodů s krokoměrem v kapse

Tabulka 15: Počet kroků při chůzi ze schodů s krokoměrem v ruce

Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	0m 56s	130	126
NXP MMA8452Q	1m 2s	130	109
Analog Devices ADXL345	1m 4s	128	114
STMicroelectronics LSM303DLH	1m 2s	128	116



(a) NXP FXOS8700CQ

(b) NXP MMA8452Q

(c) Analog Devices ADXL345

(d) STMicroelectronics LSM303DLH

Obrázek 20: Třísekundové ukázky chůze ze schodů s krokoměrem v kapse

Tabulka 16: Počet kroků při chůzi do schodů s krokoměrem v kapse

Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	1m 14s	103	108
NXP MMA8452Q	1m 20s	109	124
Analog Devices ADXL345	1m 30s	122	123
STMicroelectronics LSM303DLH	1m 18s	115	120

Tabulka 17: Počet kroků při chůzi ze schodů s krokoměrem v kapse

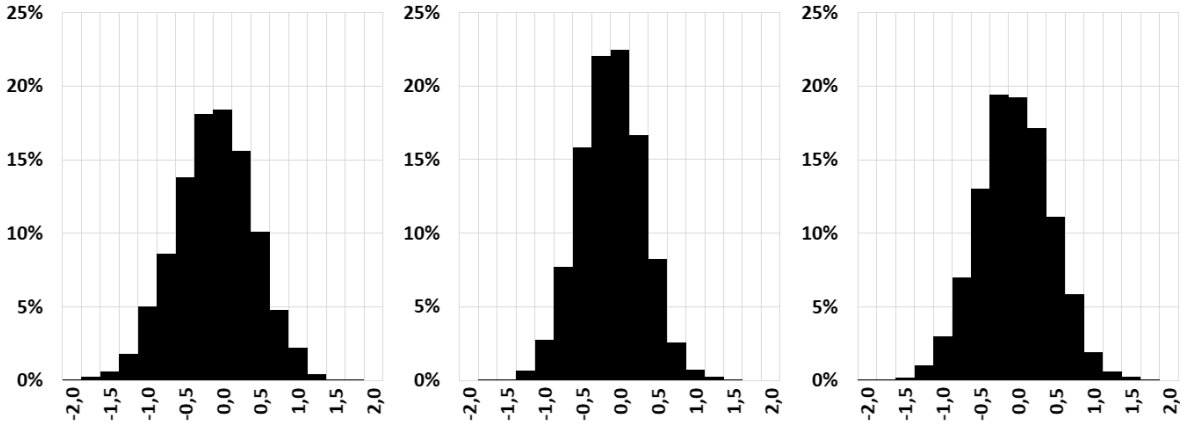
Měření s akcelerometrem	délka chůze	Xiaomi Mi Band	Nokia 6.1
NXP FXOS8700CQ	0m 59s	119	66
NXP MMA8452Q	1m 4s	131	85
Analog Devices ADXL345	1m 2s	118	14
STMicroelectronics LSM303DLH	1m 3s	128	81

6.5 Šum

Pro zjištění distribuce šumu byl napsán konzolový program `signal-distribution`, viz sekce 7.1, pro vygenerování hodnot do histogramu. Histogramy zobrazují distribuci šumu vyjádřenou v mg.

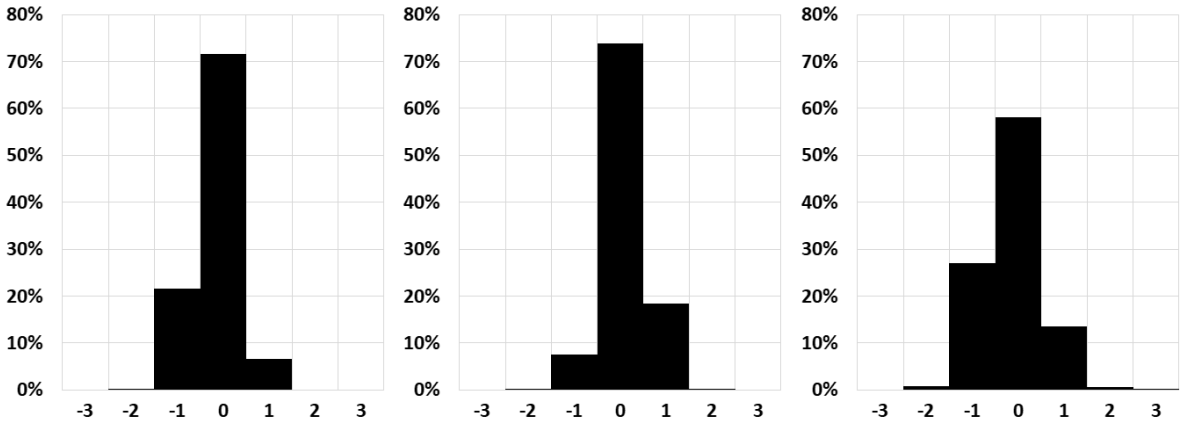
S každým akcelerometrem proběhla 10minutová měření v klidu. Samotné měření programem začíná vždy od šesté minuty a trvá dvě minuty. Důvodem je, ať se případné pnutí v kabelu při připojení externích akcelerometrů uvolní a nepůsobí posun šumu v čase. Výstupem programu je tedy distribuce naměřených hodnot z 6000 vzorků při 50 Hz.

Z důvodu, že se rozlišení a rozptyl šumu mezi jednotlivými akcelerometry výrazně liší, zobrazené výsledky v histogramech nejsou mezi jednotlivými akcelerometry přímo porovnatelné, neboť osy histogramu nezobrazují stejné rozsahy. Toto se netýká jednotlivých os akcelerometrů, kde jsou rozsahy vždy stejné.



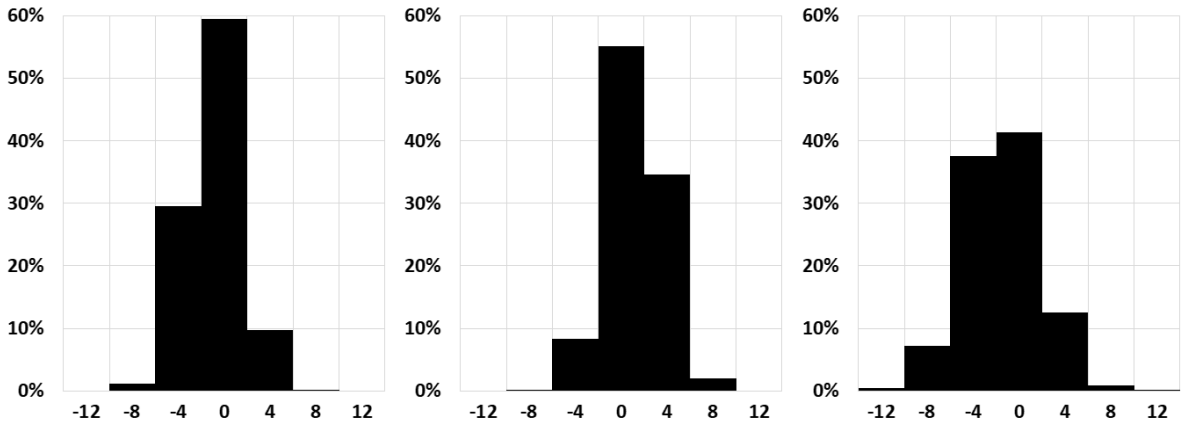
(a) osa X (b) osa Y (c) osa Z

Obrázek 21: Distribuce šumu akcelerometru NXP FXOS8700CQ



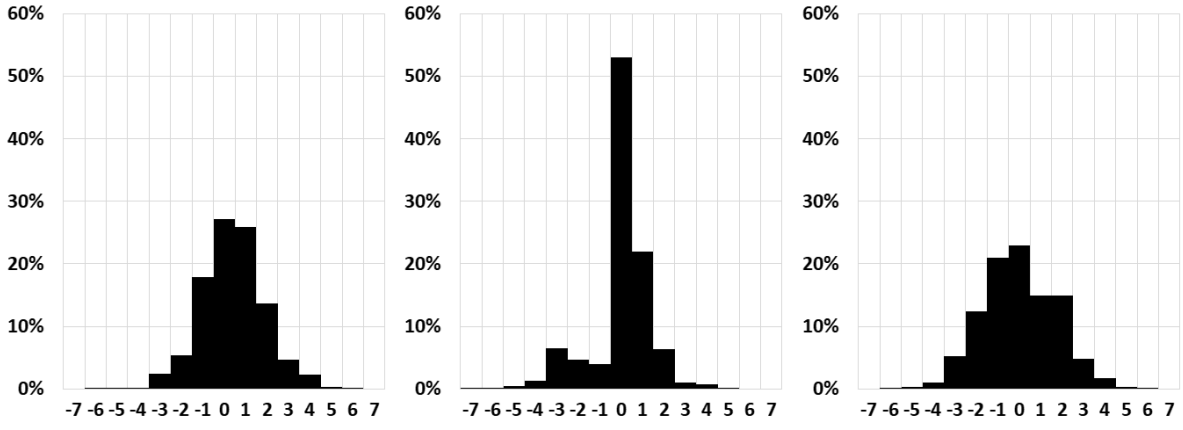
(a) osa X (b) osa Y (c) osa Z

Obrázek 22: Distribuce šumu akcelerometru NXP MMA8452Q



(a) osa X (b) osa Y (c) osa Z

Obrázek 23: Distribuce šumu akcelerometru Analog Devices ADXL345



(a) osa X (b) osa Y (c) osa Z

Obrázek 24: Distribuce šumu akcelerometru STMicroelectronics LSM303DLH

7 Programy pracující s naměřenými daty

Při vývoji byly vytvořeny dva konzolové programy v jazyce C. Prvním programem je program `signal-distribution`, který je určen pro získání distribuce naměřených dat akcelerometry na všech třech osách. Tento program byl použit v sekci 6.5 pro získání distribuce šumu jednotlivých akcelerometrů. Druhým programem je program `step-analyzer`, který slouží pro testování implementovaných filtrů popsaných v sekci 8 a pro testování algoritmů detekce kroků popsaných v sekci 9.

Oba programy spolu sdílejí způsob, jakým připravují data pro jejich další zpracování. Pro každý kanál vstupního WAV souboru je alokováno potřebné místo. Data jsou poté rozdělena do jednotlivých kanálů, viz výpis 8.

```
for(int i = 0; i < Subchunk2Size/sizeof(*inputData); i++) {
    inputChannels[i%NumChannels][i/NumChannels] = inputData[i];
}
```

Výpis 8: Rozdělení jednotlivých kanálů

7.1 Program pro měření distribuce dat

Tento program funguje tím způsobem, že pro každou osu spočítá počet výskytů hodnot v signálu, viz výpis 9. Výsledky jsou vytisknuty programem do konzole. Program přijímá 3 parametry v následujícím pořadí:

1. Cesta k WAV souboru.
2. Počáteční sekunda měření.
3. Délka měření v sekundách.

```
for(int i = secondsStart*SampleRate; i < secondsStart*SampleRate +
    secondsLength*SampleRate; i++) {
    px[inputChannels[0][i]]++;
    py[inputChannels[1][i]]++;
    pz[inputChannels[2][i]]++;
}
```

Výpis 9: Měření distribuce hodnot

7.2 Program pro měření distribuce dat

K testování filtrů a algoritmů detekce kroků byl vytvořen konzolový program `step-analyzer`. Program přijímá 3 parametry v následujícím pořadí:

1. Cesta k WAV souboru.
2. Počáteční sekunda měření detekce chůze.
3. Délka měření detekce chůze v sekundách.

Vstupem tohoto programu je WAV soubor vytvořený vývojovou deskou, přesněji jeho 4. kanál, na kterém je naměřená velikost tíhového zrychlení g.

Výstupem tohoto programu jsou dva soubory. Tím prvním je WAV soubor, kde jsou liché kanály určeny jednotlivým filtrům, které jsou popsány v sekci 8. Filtrování probíhá na celém signálu bez rozdílu, jestli je měření omezeno, nebo ne. Na prvním kanálu je původní nefiltrovaný signál. Sudé kanály jsou určeny pro značení detekce kroků, které jsou popsány v sekci 9. Druhým souborem je textový soubor, ve kterém jsou napočítány počty kroků oběma implementovanými algoritmy detekce kroků, a to pro všechny implementované filtry.

Jméno výstupního WAV a TXT souboru je jméno vstupního souboru s připojeným `-filtered` na konci.

Program pro každý výstupní kanál alokuje místo, nad kterými pracují jednotlivé filtry nebo algoritmy detekce kroků. Po filtrování a detekci kroků je k WAV souboru vytvořená hlavička a jednotlivé kanály filtrovaného signálu jsou seřazeny a uloženy ve správném formátu.

```
for(int i = 0; i < outputSubchunk2Size/sizeof(*inputData); i++) {
    outputData[i] = outputChannels[i%NUM_OF_CHANNELS][i/NUM_OF_CHANNELS];
}
```

Výpis 10: Spojení jednotlivých kanálů do souboru WAV

8 Filtrování signálu

Úkolem filtrování naměřeného signálu je odstranění šumu. To je potřeba pro snazší detekci chůze. Normální chůze málokdy překračuje rychlost čtyř kroků za sekundu, a proto frekvence vyšší než 4 Hz lze považovat za šum. Při vyhledávání vhodných filtrů byly hledány filtry typu dolní propust, které by byly schopny odstranit šum.

Tato sekce popisuje principy použitých filtrů a jejich porovnání. Implementace samotných filtrů byla inspirována knihou The Scientist & Engineer's Guide to Digital Signal Processing [15]. Všechny níže popsané filtry jsou součástí implementace programu `step-analyzer`, viz sekce 7.2, který aplikuje tyto filtry na signál naměřený při chůzi a výsledky uloží do WAV souboru.

8.1 Filtry s konečnou impulzní odezvou

Výstup filtru s konečnou impulzní odezvou je definován jako konvoluční suma:

$$y[i] = (h * x)[i] = \sum_{j=0}^{M-1} x[i-j]h[j] \tag{2}$$

kde $y[i]$ je výstupní hodnota, $x[i]$ je vstupní hodnota, $h[j]$ je hodnota jádra a M je velikost jádra.

8.1.1 Klouzavý průměr

Tento filtr funguje na principu průměrování určitého počtu vzorků vstupního signálu pro vytvoření každého vzorku výstupního signálu. Tento filtr je definován:

$$y[i] = \frac{1}{M} \sum_{j=0}^{M-1} x[i-j] \tag{3}$$

kde $y[i]$ je výstupní hodnota, $x[i]$ je vstupní hodnota a M je počet průměrovaných vzorků. Tento filtr je konvolucí o velikosti jádra M a hodnotami v jádře $1/M$.

Implementace tohoto filtru byla zjednodušena způsobem, kdy je uložen celkový součet průměrovaného úseku a při výpočtu následující výstupní hodnoty se odečte vstupní hodnota posledního vzorku a přičte se vstupní hodnota nového vzorku. Výsledná hodnota se pak vydělí počtem průměrovaných vzorků. Výhodou tohoto přístupu je, že rychlost výpočtu je vždy stejná bez ohledu na velikost jádra.

Při filtrování signálu klouzavým průměrem byla po experimentech zvolena velikost jádra M jako 1/6 vzorkovací frekvence, tedy při 50 Hz je $M = 8$. Výsledky tohoto filtru jsou uloženy na 3. kanále filtrovaného WAV souboru.

8.1.2 Konvoluce s funkcí sinc

Sinc funkce při použití v konvoluci se signálem je definována jako:

$$h[i] = \frac{\sin(2\pi f_c i)}{i\pi} \tag{4}$$

kde $h[i]$ je hodnota jádra a f_c je mezní frekvence vyjádřena jako podíl vůči vzorkovací frekvenci.

Problém je, že obor hodnot sinc funkce je nekonečný, a tak je potřeba jej omezit tak, aby se vešla do jádra o velikosti M. Při omezení ale na koncích jádra vznikne náhle přerušení, které je potřeba vyhladit. Pro vyhlazení bylo použito Blackmanovo okno. Pro lehčí implementaci je funkce posunutá o $M/2$, aby jádro mohlo být indexováno pouze kladnými hodnotami. Jádro filtru je vypočteno z funkce:

$$h[i] = K\frac{\sin(2\pi f_c(i - M/2))}{i - M/2}\left[0.42 - 0.5\cos\left(\frac{2\pi i}{M}\right) + 0.08\cos\left(\frac{4\pi i}{M}\right)\right] \tag{5}$$

kde $h[i]$ je hodnota jádra, f_c je mezní frekvence vyjádřena jako podíl vůči vzorkovací frekvenci a konstanta K složí pro normalizaci jádra. Index i běží od nuly po M včetně, a tak velikost výsledného jádra, které musí být liché je $M + 1$. Pokud je index i roven $M/2$, tak je použit:

$$h[i] = 2\pi f_c K \tag{6}$$

Při filtrování signálu konvolucí s funkci sinc byla zvolena mezní frekvence 4 Hz, což při vzorkovací frekvenci 50 Hz dává hodnotu $f_c = 0,08$. Velikost jádra $M+1$ byla zvolena jako 1/2 vzorkovací frekvence, tedy při 50 Hz je $M + 1 = 25$. Výsledky tohoto filtru jsou uloženy na 5. kanále filtrovaného WAV souboru.

8.2 Filtry s nekonečnou impulzní odezvou

Na rozdíl od filtrů s konečnou impulzní odezvou, které k výpočtu používají pouze hodnoty ze vstupního signálu, tyto filtry používají i hodnoty z výstupního signálu. Výstup filtru s nekonečnou impulzní odezvou je definován jako:

$$y[n] = \sum_{i=0}^{j} a_i x[n-i] + \sum_{i=1}^{k} b_i y[n-i] \tag{7}$$

kde y[n] je výstupní hodnota, x[n] je vstupní hodnota a hodnoty a_i a b_i jsou koeficienty daného filtru.

8.2.1 Jednopólová dolní propust

Tento filtr má pouze dva koeficienty, a to:

$$\begin{aligned} a_0 &= 1 - x \\ b_1 &= x \end{aligned} \tag{8}$$

kde x je hodnotou mezi nulou a jedna, která kontroluje sílu filtru.

Při použití tohoto filtru byla hodnota x zvolena jako $x = 0,85$, což dává jednotlivým koeficientům hodnoty $a_0 = 0,15$ a $b_1 = 0,85$. Výsledky tohoto filtru jsou uloženy na 7. kanále filtrovaného WAV souboru.

8.2.2 Čtyřstupňová dolní propust

Tento filtr je složen z pěti koeficientů, a to:

$$\begin{aligned} a_0 &= (1 - x)^4 \\ b_1 &= 4x \\ b_2 &= -6x^2 \\ b_3 &= 4x^3 \\ b_4 &= -x^4 \end{aligned} \tag{9}$$

kde x je hodnotou mezi nulou a jedna, která kontroluje sílu filtru.

Při použití tohoto filtru byla hodnota x zvolena jako $x = 0,6$. Výsledky tohoto filtru jsou uloženy na 9. kanále filtrovaného WAV souboru.

8.2.3 Dvoupólový Čebyševův filtr

Tento filtr se skládá z celkem pěti koeficientů. Tyto koeficienty byly převzaty z tabulek hodnot pro dvoupólový Čebyševův filtr knihy [15]. Byly zvoleny koeficienty pro mezní frekvenci $f_c = 0,075$.

$$\begin{aligned} a_0 &= \texttt{3.869430E-02} \\ a_1 &= \texttt{7.738860E-02} \\ a_2 &= \texttt{3.869430E-02} \end{aligned} \qquad \begin{aligned} b_1 &= \texttt{1.392667E+00} \\ b_2 &= \texttt{-5.474446E-01} \end{aligned} \tag{10}$$

Výsledky tohoto filtru jsou uloženy na 11. kanále filtrovaného WAV souboru.

8.2.4 Čtyřpólový Čebyševův filtr

Tento filtr se skládá z celkem devíti koeficientů. Tyto koeficienty byly převzaty z tabulek hodnot pro čtyřpólový Čebyševův filtr knihy [15]. Byly zvoleny koeficienty pro mezní frekvenci

$f_c = 0,075$

$$\begin{aligned} a_0 &= \texttt{9.726342E-04} & b_1 &= \texttt{3.103944E+00} \\ a_1 &= \texttt{3.890537E-03} & b_2 &= \texttt{-3.774453E+00} \\ a_2 &= \texttt{5.835806E-03} & b_3 &= \texttt{2.111238E+00} \\ a_3 &= \texttt{3.890537E-03} & b_4 &= \texttt{-4.562908E-01} \\ a_4 &= \texttt{9.726342E-04} \end{aligned} \tag{11}$$

Výsledky tohoto filtru jsou uloženy na 13. kanále filtrovaného WAV souboru.

8.3 Výpočetní náročnost vybraných filtrů

Pro měření výpočetní náročnosti filtrů, byly filtry implementovány na vývojovou desku NXP FRDM-K64F. Filtry které používaly výpočty s plovoucí desetinnou čárkou byly implementovány s použitím výpočtů s jednoduchou přesností. Ke zjištění výpočetní náročnosti byl využit systick časovač, který je součástí jádra mikrokontroléru. Jedná se o 24bitový časovač který počítá od stanovené hodnoty do nuly. Tabulka 18 zobrazuje počet tiků potřebných pro výpočet jednoho vzorku výstupního signálu, a tedy jejich náročnost na výpočet.

Tabulka 18: Výpočetní náročnost vybraných filtrů

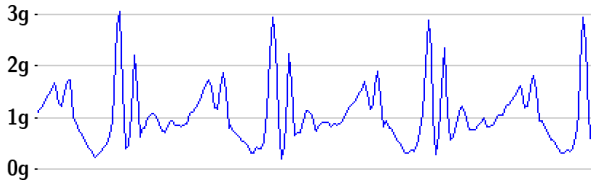
Filtr	počet tiků
Klouzavý průměr	23
Konvoluce s funkcí sinc	308
Jednopólová dolní propust	51
Čtyřstupňová dolní propust	103
Dvoupólový Čebyševův filtr	87
Čtyřpólový Čebyševův filtr	200

Experimentem bylo dokázáno, že rychlost tiku systick časovače je shodná s rychlostí procesoru, který je 120 MHz. Při experimentu byl časovač nastaven na svou maximální hodnotu `0xFFFFFF`. Při dosažení nuly systick vyvolal přerušení, které přepínalo RGB LED pro signalizaci vyvolání přerušení. Těchto bliknutí RGB LED bylo přibližně 3,5 za sekundu. Jelikož každé přerušení pouze přepnulo RGB LED, to odpovídá přibližně 7 přerušením za sekundu. Teoreticky by rychlost vyvolání přerušení měla být 120 MHz/`0xFFFFFF` $\approx$ 7,15 což přibližně odpovídá hodnotě získané při experimentu.

8.4 Porovnání vybraných filtrů

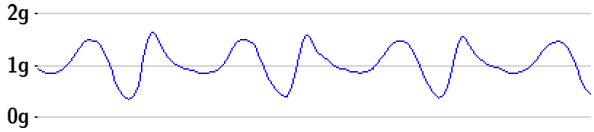
Obrázek 25 zobrazuje výstupy jednotlivých filtrů aplikovaných na signál naměřený při chůzi po rovině a do kopce s akcelerometrem Analog Devices ADXL345 umístěným v kapse. Při porovnání filtrů zaujmou svou kvalitou především dva filtry. Těmi jsou filtry konvoluce s funkcí sinc a

čtyřpólový Čebyševův filtr. Výhodou těchto dvou filtrů je jejich schopnost odstranit velké množství šumu a zachování amplitudy signálu. Nevýhodou je jejich výpočetní náročnost. Dvoupólový Čebyševův filtr se zdá být také jako vhodný, neboť nabízí kompromis mezi výpočetní náročností a schopnosti odstranit šum.

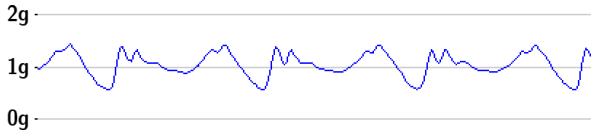


(a) Nefiltrovaný signál

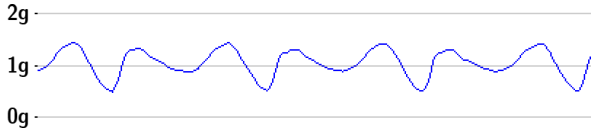
(b) Klouzavý průměr



(c) Konvoluce s funkcí sinc

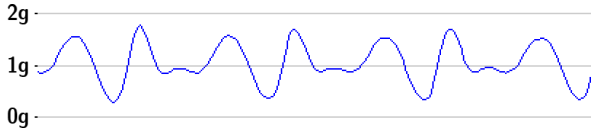


(d) Jednopólová dolní propust



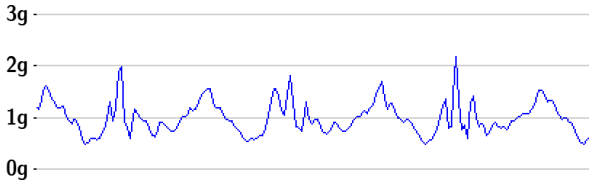
(e) Čtyřstupňová dolní propust

(f) Dvoupólový Čebyševův filtr

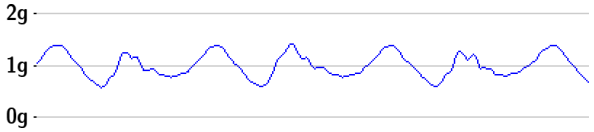


(g) Čtyřpólový Čebyševův filtr

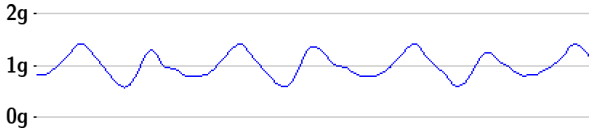
Chůze po rovině s akcelerometrem v kapse



(a) Nefiltrovaný signál

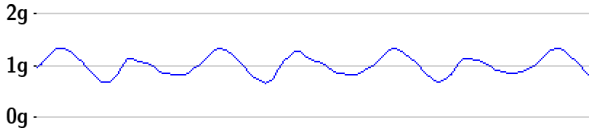


(b) Klouzavý průměr

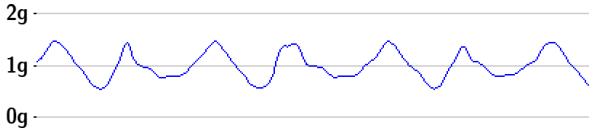


(c) Konvoluce s funkcí sinc

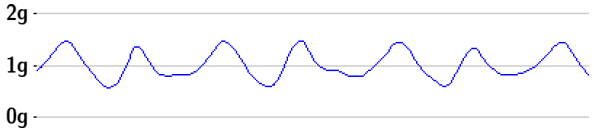
(d) Jednopólová dolní propust



(e) Čtyřstupňová dolní propust



(f) Dvoupólový Čebyševův filtr



(g) Čtyřpólový Čebyševův filtr

Chůze do kopce s akcelerometrem v kapse

Obrázek 25: Čtyřsekundové ukázky aplikovaných filtrů

9 Detekce kroků

Byly implementovány 2 algoritmy pro detekci a počítání kroků. Algoritmy jsou součástí konzolového programu `step-analyzer`, který již byl popsán v sekci 7.2. Algoritmy detekce chůze byly aplikovány na výsledky všech implementovaných filtrů. Výstupem tohoto programu je WAV soubor s lichými kanály věnovanými filtrům a sudými kanály pro značení detekce kroků. Značení v jednotlivých kanálech dosahuje celkem 4 hladin:

1. 0/3 - žádný algoritmus nedetekoval krok
2. 1/3 - první z popsaných algoritmů detekoval krok
3. 2/3 - druhý z popsaných algoritmů detekoval krok
4. 3/3 - oba z popsaných algoritmů detekovaly krok

Toto značení u obou algoritmů nastane až po 5 úspěšně naměřených krocích. Dále tento program generuje textový TXT soubor se stejným názvem jako vstupní soubor s připojeným `-filtered` na konci jména, ve kterém jsou napsány počty naměřených kroků jednotlivými algoritmy detekce kroků pro všechny filtrované signály.

Experimenty bylo zjištěno, že při lehké chůzi po koberci, je rozdíl mezi minimem a maximem na naměřeném signálu přibližně 4000. To odpovídá hodnotě přibližně 244 mg. S touto hodnotou se tedy počítalo jako hodnotou, kdy má smysl detekovat krok. V níže popsaných algoritmech se o této hodnotě mluví jako o rozlišovací schopnosti.

9.1 První způsob detekce kroků

Tento první způsob detekce a počítání kroků byl převzat s úpravami z popisu algoritmu z článku Full-Featured Pedometer Design Realized with 3-Axis Digital Accelerometer [16]. Na rozdíl od tohoto článku, kde algoritmus pracuje s hodnotami z osy, na které byl naměřen nejsilnější signál, tento algoritmus pracuje s celkovou velikostí tíhového zrychlení.

Tento algoritmus každou sekundu měří maximální a minimální hodnotu na signálu, a podle těchto hodnot upravuje svou rozlišovací schopnost `precision` a práh `threshold` pro další sekundu. Práh je průměrem maxima a minima a slouží pro detekci kroků. Rozlišovací schopnost `precision` slouží pro filtrování vstupního signálu.

Proměnná `sample_result` v sobě ukládá aktuální hodnotu filtrovaného signálu. Proměnná `sample_old` slouží pro uložení předchozího stavu proměnné `sample_new`, která v sobě ukládá hodnotu z proměnné `sample_result`. Ta se však mění jen tehdy, je li rozdíl mezi ní a proměnnou `sample_result` větší, než je rozlišovací schopnost `precision`. Krok je pak počítán v momentě, kdy signál klesá a `sample_new` klesne pod definovaný práh, ale `sample_old` je stále nad ním.

Tento způsob ale zachycuje i různé náhodné pohyby. Proto je toto řešení rozšířeno o omezení, kdy je potřeba udělat alespoň 5 kroků, než začne ostré počítání. Veškeré kroky musí navíc být

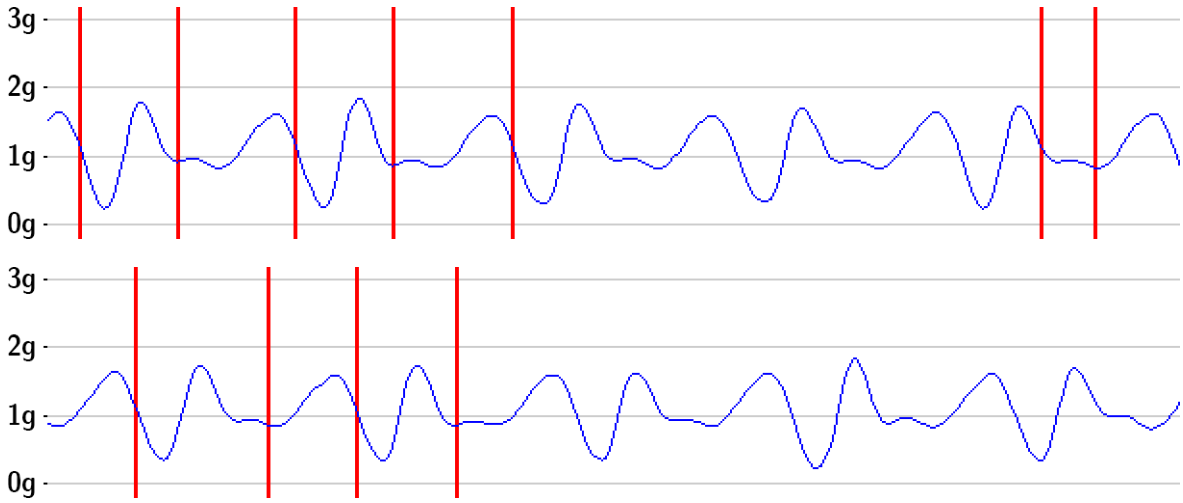
v rozmezí 1/4 sekundy až 2 sekund.

```
for(int i = secondsStart*SampleRate; i < secondsStart*SampleRate +
    secondsLength*SampleRate; i++) {
    sample_result = outputChannels[ch][i];
    sample_old = sample_new;
    sample_new = (abs(sample_new-sample_result)>precision)?sample_result:
        sample_new;
    if(i%(SampleRate) == 0) {
        threshold = (min + max) / 2;
        precision = (max-min>8000)?(max-min)/4:2000;
        max = INT_MIN;
        min = INT_MAX;
    }
    max = (sample_result>max)?sample_result:max;
    min = (sample_result<min)?sample_result:min;
    if(sample_new <= threshold && threshold <= sample_old) {
        if(i-last_step_sample > SampleRate*2) {
            mode = 0;
            temporary_steps = 1;
        }
        switch(mode) {
        case 0:
            if(i-last_step_sample > SampleRate/4) {
                if(++temporary_steps >= 5) {
                    mode = 1;
                    total_steps += temporary_steps;
                    temporary_steps = 0;
                }
            }
            else {
                temporary_steps = 0;
            }
            break;
        case 1:
            if(i-last_step_sample > SampleRate/4) {
                total_steps++;
                outputChannels[ch+1][i] += INT16_MAX/3;
            }
```

```
            else
                mode = 0;
            break;
        }
        last_step_sample = i;
    }
}
```

Výpis 11: Implementace prvního způsobu detekce kroků

Při testování se ale ukázala nevýhoda tohoto řešení detekce kroků a myšlenky použití prahu pro detekci kroků. Obrázek 26 zobrazuje selhání tohoto způsobu detekce kroků, kdy signál pravděpodobně nepřekročil onen práh a krok nebyl detekován. Červené vertikální čáry značí, kdy byl krok detekován.



Obrázek 26: Příklady selhání prvního způsobu detekce kroků

9.2 Vlastní řešení detekce kroků

Toto řešení funguje na principu hledání nástupních a sestupných hran na signálu. Algoritmus implementuje stavový automat, který má dva stavy. Prvním stavem je hledáni nástupní hrany, kdy je každý vstupní vzorek porovnán s minimem, a pokud je menší, tak je uložen. Jakmile je vstupní vzorek o rozlišovací schopnost větší než než nalezené minimum, přechází automatu do druhého stavu. Druhý stav je určen pro hledání sestupné hrany, kdy je každý vstupní vzorek porovnán s maximem, a pokud je větší, tak je uložen. Jakmile je vstupní vzorek o rozlišovací schopnost menší než nalezené maximum, zaznamená se krok, a automat přechází do prvního stavu.

Stejně jako v předchozím případě, i zde je implementováno omezení, kdy je potřeba udělat alespoň 5 kroků než začne ostré počítání, a kdy veškeré kroky musí být v rozmezí 1/4 sekundy až 2 sekund.

Tento algoritmus je dále rozšířen o dynamickou rozlišovací schopnost, která závisí na dynamice chůze. Při normální chůzi je potřeba pro přičtení kroku, aby signál klesl alespoň o 1/3 síly signálu. Při porovnání s předchozím algoritmem, se tento způsob detekce kroků ukázal jako spolehlivější.

```
for(int i = secondsStart*SampleRate; i < secondsStart*SampleRate +
    secondsLength*SampleRate; i++) {
    sample_result = outputChannels[ch][i];
    if(i-last_step_sample > SampleRate*2) {
        precision = 4000;
        temporary_steps = 0;
        mode = 0;
    }
    switch(state) {
    case 0:
        min = (sample_result<min)?sample_result:min;
        if(sample_result - min > precision) {
            state = 1;
            precision = precision*0.8f + ((max-min>12000)?(max-min)/3:4000)*0.2f
                ;
            max = sample_result;
        }
        break;
    case 1:
        max = (sample_result>max)?sample_result:max;
        if(max - sample_result > precision) {
            state = 0;
            precision = precision*0.8f + ((max-min>12000)?(max-min)/3:4000)*0.2f
                ;
            if(i-last_step_sample > SampleRate/4) {
                switch(mode) {
                case 0:
                    if(++temporary_steps >= 5) {
                        mode = 1;
                        total_steps += temporary_steps;
                    }
```

```
                    break;
                case 1:
                    total_steps++;
                    outputChannels[ch+1][i] += (INT16_MAX/3)*2;
                    break;
                }
            }
            else {
                temporary_steps = 0;
                mode = 0;
            }
            last_step_sample = i;
            min = sample_result;
        }
        break;
    }

}
```

Výpis 12: Implementace vlastního způsobu detekce kroků

9.3 Porovnání počtů kroků s dostupnými krokoměry

Následující část se zabývá porovnáním počtu kroků mezi krokoměry a všemi kombinacemi filtrů a algoritmů detekce kroků. Obrázky 27, 28, 29 a 30 zobrazují počty kroků, kdy označení a1 a a2 označují první a druhy způsob detekce kroků, a označení f1 až f6 označují jednotlivé filtry, jmenovitě:

1. Klouzavý průměr.
2. Konvoluce s funkcí sinc.
3. Jednopólová dolní propust.
4. Čtyřstupňová dolní propust.
5. Dvoupólový Čebyševův filtr.
6. Čtyřpólový Čebyševův filtr.

	krokoměry		f1		f2		f3		f4		f5		f6	
	Mi Band	Nokia 6.1	a1	a2	a1	a2	a1	a2	a1	a2	a1	a2	a1	a2
rovina-ruka	1165	1176	1147	1183	1142	1184	1125	1177	1119	1178	1158	1187	1167	1187
rovina-kapsa	1168	1191	1113	1188	1016	1188	1108	1033	1080	1188	1078	1193	965	1188
tráva-ruka	768	787	767	780	768	782	732	721	756	761	770	783	771	782
tráva-kapsa	728	763	686	775	619	775	690	773	660	774	647	750	672	781
kopec-ruka-nahoru	345	351	347	343	344	343	339	343	336	343	342	344	339	344
kopec-ruka-dolu	325	325	332	323	330	323	329	324	324	324	323	328	328	328
kopec-kapsa-nahoru	352	350	304	352	291	352	289	340	285	332	285	353	302	353
kopec-kapsa-dolu	322	320	330	327	324	327	308	326	313	325	337	330	325	327
schody-ruka-nahoru	112	106	116	119	123	122	99	95	111	101	118	126	119	126
schody-ruka-dolu	130	126	121	128	120	128	117	115	119	119	126	129	120	128
schody-kapsa-nahoru	103	108	96	110	97	125	86	95	73	84	102	128	104	127
schody-kapsa-dolu	119	66	87	99	86	91	84	89	81	87	93	89	92	96

Obrázek 27: Porovnání počtu kroků s akcelerometrem NXP FXOS8700CQ

	krokoměry		f1		f2		f3		f4		f5		f6	
	Mi Band	Nokia 6.1	a1	a2	a1	a2	a1	a2	a1	a2	a1	a2	a1	a2
rovina-ruka	1187	1185	1179	1186	1189	1191	1105	1166	1140	1183	1201	1198	1201	1198
rovina-kapsa	1192	1192	1012	1199	838	1200	940	850	967	1192	949	1201	854	1201
tráva-ruka	729	759	739	775	740	773	653	562	705	702	744	787	758	784
tráva-kapsa	724	770	687	786	640	783	627	772	616	776	708	785	707	788
kopec-ruka-nahoru	339	337	339	339	342	339	334	339	339	339	346	339	345	339
kopec-ruka-dolu	316	315	315	317	316	317	306	316	308	317	315	316	319	316
kopec-kapsa-nahoru	347	345	314	347	305	347	290	345	300	344	327	347	322	347
kopec-kapsa-dolu	325	322	289	324	294	324	275	323	278	324	283	324	292	324
schody-ruka-nahoru	128	132	132	135	130	135	128	130	130	133	130	135	136	134
schody-ruka-dolu	130	109	118	126	115	127	109	108	114	111	120	128	120	129
schody-kapsa-nahoru	109	124	114	137	120	136	98	120	99	118	120	137	125	137
schody-kapsa-dolu	131	85	127	123	117	123	111	117	114	121	123	125	124	126

Obrázek 28: Porovnání počtu kroků s akcelerometrem NXP MMA8452Q

	krokoměry		f1		f2		f3		f4		f5		f6	
	Mi Band	Nokia 6.1	a1	a2	a1	a2	a1	a2	a1	a2	a1	a2	a1	a2
rovina-ruka	1200	1185	1185	1188	1180	1188	1157	1187	1179	1188	1185	1188	1187	1188
rovina-kapsa	1191	1182	1001	1191	911	1190	968	554	885	1185	976	1202	845	1205
tráva-ruka	744	774	742	779	732	779	639	672	707	770	751	780	752	779
tráva-kapsa	710	738	671	673	672	685	638	651	635	638	695	720	679	703
kopec-ruka-nahoru	343	339	343	342	343	342	343	342	342	342	340	342	343	342
kopec-ruka-dolu	319	318	317	320	318	319	306	319	307	319	320	319	320	320
kopec-kapsa-nahoru	348	347	334	339	321	330	312	299	305	299	342	347	342	352
kopec-kapsa-dolu	326	319	298	330	277	331	270	323	272	321	270	331	287	334
schody-ruka-nahoru	125	122	127	129	127	129	123	123	127	128	127	128	129	128
schody-ruka-dolu	128	114	108	118	115	121	106	101	113	110	116	123	115	123
schody-kapsa-nahoru	122	123	104	129	119	129	92	99	98	103	117	130	104	129
schody-kapsa-dolu	118	14	92	90	93	91	75	86	86	87	95	99	98	96

Obrázek 29: Porovnání počtu kroků s akcelerometrem Analog Devices ADXL345

	krokoměry		f1		f2		f3		f4		f5		f6	
	Mi Band	Nokia 6.1	a1	a2	a1	a2	a1	a2	a1	a2	a1	a2	a1	a2
rovina-ruka	1183	1183	1181	1183	1179	1183	1182	1183	1182	1183	1184	1184	1182	1183
rovina-kapsa	1201	1201	1111	1202	941	1202	1081	1202	1055	1202	1011	1202	1010	1202
tráva-ruka	726	724	722	726	722	726	719	725	718	725	720	726	721	726
tráva-kapsa	715	709	632	715	548	715	612	714	603	714	608	714	602	714
kopec-ruka-nahoru	338	336	337	339	337	339	338	337	337	338	338	339	336	339
kopec-ruka-dolu	321	318	321	320	320	320	317	320	321	320	319	320	320	320
kopec-kapsa-nahoru	347	337	316	354	270	344	271	339	281	340	279	358	305	358
kopec-kapsa-dolu	330	326	323	327	321	327	306	327	314	327	307	327	315	332
schody-ruka-nahoru	127	119	124	130	125	130	117	118	125	123	129	130	128	130
schody-ruka-dolu	128	116	121	120	115	120	108	106	113	110	117	125	116	125
schody-kapsa-nahoru	115	120	113	124	99	122	87	96	79	81	121	128	121	127
schody-kapsa-dolu	128	81	109	112	106	112	91	95	96	94	119	122	114	117

Obrázek 30: Porovnání počtu kroků s akcelerometrem STMicroelectronics LSM303DLH

10 Implementace krokoměru na desce NXP FRDM-K64F

Závěrečná implementace krokoměru používá pro filtrování naměřeného signálu dvoupólový Čebyševův filtr. Pro detekci a počítání kroků byl zvolen autorův vlastní algoritmus.

10.1 Ovládání krokoměru na desce NXP FRDM-K64F

Vývojová deska NXP FRDM-K64F je osazena 2 tlačítky a RGB LED. Ty byly využity pro jednoduché ovládání a signalizaci stavu. Obě tlačítka jsou připojena pull-up rezistory k V_{CC} a jejich stiskem je signál na nich přiveden ke GND. Tlačítko `SW2` je připojeno k pinu mikrokontroléru `PTC6` a tlařítko `SW3` k pinu `PTA4`.

Při připojení vývojové desky k napájení, se vyhledá připojený akcelerometr. Pokud je připojen externí akcelerometr, RGB LED se rozsvítí tyrkysovou barvou. V případě, kdy není připojen externí akcelerometr, rozsvítí se RGB LED modře. Tyrkysová/modrá barva tedy indikuje správnou inicializaci akcelerometrů.

Z toho důvodu, že deska je při některých měřeních a experimentech nošena v kapse, bylo implementováno opatření nechtěného stisku tlačítka. Obě tyto tlačítka jsou v software propojena tak, aby se chovala jako jedno. Pro registraci stisku tlačítka je potřeba, aby obě tlačítka byla stisknutá. Pro registraci uvolnění tlačítka je potřeba, aby obě tlačítka byla uvolněna.

Po vyhledání a inicializaci připojeného akcelerometru se zařízení stisknutím obou tlačítek přepíná mezi 2 stavy:

1. Pohotovostní režim - signalizován tyrkysovou či modrou barvou (podle připojeného akcelerometru). V tomto režimu je možné připojit vývojovou desku k počítači a dostat z ní naměřené data. Samotné měření dat neprobíhá.

2. Režim měření - signalizován blikáním modré RGB LED při zápisu dat na SD kartu, a blikáním zelené RGB LED při chůzi. V tomto režimu je implementovaný krokoměr aktivní a probíhá ukládání dat na SD kartu.

Pokud nastane jakákoliv chyba, rozsvítí se RGB LED červeně.

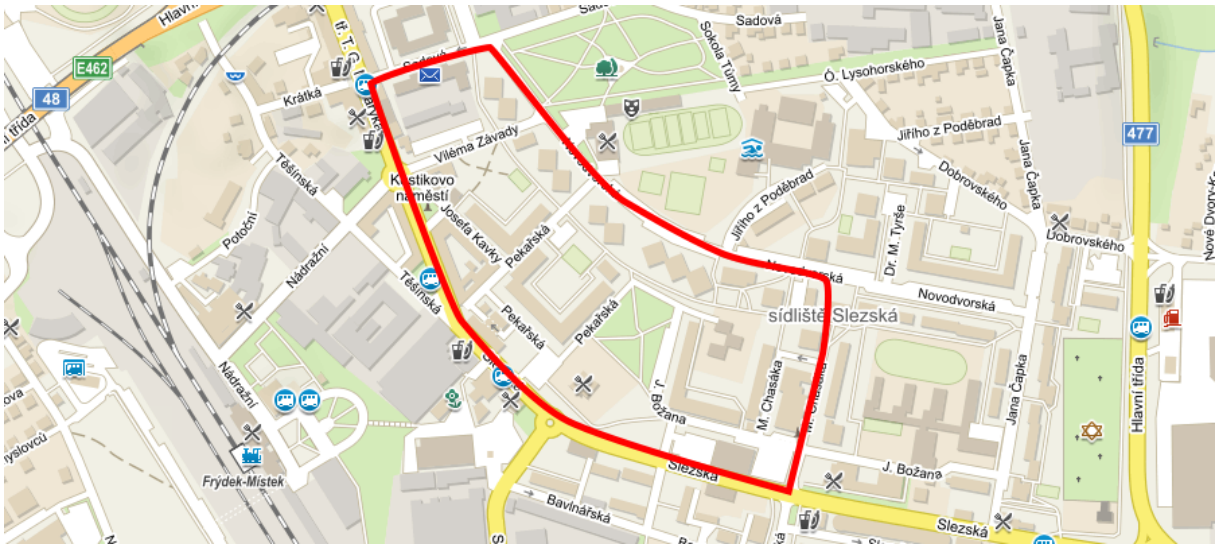
Pro naformátováni připojené SD karty je potřeba při přívodu napájení držet tlačítko `SW2`. V momentě, kdy se rozsvítí žlutá RGB LED je potřeba stisknout obě tlačítka `SW2` a `SW3` najednou a následně je uvolnit. Po uvolnění tlačítek žlutá RGB LED zhasne a započne formátování připojené SD karty. Formátování započne až po zhasnutí žluté RGB LED, a tak má uživatel čas na to se rozhodnout, zda chce danou kartu naformátovat.

V případě, že se nepodaří mikrokontrolérem rozpoznat souborový systém SD karty, rozsvítí se žlutá RGB LED pro možnost naformátováni připojené SD karty. Stisknutím obou tlačítek a jejich následným uvolněním žlutá RGB LED zhasne a započne samotné formátování.

10.2 Závěrečné testování chůzí po městě

Závěrečné testování kompletního řešení probíhalo chůzi po městě. Z toho důvodu, že se signál chůze výrazně neliší mezi akcelerometry, se následující část zabývá testováním pouze s akcelerometrem NXP FXOS8700CQ, který je součástí desky NXP FRDM-K64F. Měřená trasa byla přibližně 1650 metrů dlouhá. Obrázek 31 zobrazuje okruh, na kterém se měření provádělo. Přibližné souřadnice zvoleného okruhu jsou 49.6790N, 18.3594E.

Tabulka 19 zobrazuje počty naměřených kroků jednotlivými krokoměry. Při porovnání počtu kroků lze vidět, že hodnoty si jsou velice blízko. Ve všech případech je rozdíl v počtu kroků mezi jednotlivými krokoměry menší než 2%.



Obrázek 31: Úsek, na kterém probíhalo závěrečné měření

Tabulka 19: Počet kroků při závěrečném měření

Umístění krokoměru	délka chůze	Xiaomi Mi Band	Nokia 6.1	FRDM-K64F
V kapse	18m 46s	1986	2013	2021
V ruce	18m 36s	1993	2007	2010

11 Závěr

Cílem této bakalářské práce bylo vytvořit krokoměr pomocí vývojového kitu s ARM procesorem.

Krokoměr byl vyvíjen na vývojové desce NXP FRDM-K64F. Byly vybrány a porovnány celkem čtyři různé akcelerometry. Naměřeny byly data z chůze po asfaltu na rovině, pomalé chůze po trávě na rovině, chůze po zpevněné lesní cestě do kopce a z kopce a chůze do schodů a ze schodů. Porovnáním naměřeného signálu vybraných akcelerometrů mezi sebou bylo zjištěno, že se chůze na všech akcelerometrech projevuje velmi podobným a pro rozpoznání kroků dostačujícím způsobem.

Při testování filtrů pro odstranění šumu se ukázalo, že výpočetně náročnější filtry podávají lepší výsledky. Kvůli tomu, že výsledná aplikace je určena pro zařízení s nízkým výkonem, byl použit dvoupólový Čebyševův filtr, který nabízí kompromis mezi výpočetní náročností a schopnosti odstranit šum.

Pro počítání kroků byly implementovány dva způsoby detekce kroků. První z nich se ukázal jako nevyhovující, neboť nedokázal v jistých případech správně detekovat kroky. Druhý způsob detekce kroků byl o poznáni přesnější.

Celé toto řešení bylo následně implementováno na vývojovou desku a otestováno chůzí po městě s doprovodem dalších dvou krokoměrů. Testováním se ukázalo, že výsledky si jsou velice podobné s výsledky ostatních dvou krokoměrů.

Literatura

[1] WEINBERG, Harvey. *Using the ADXL202 in Pedometer and Personal Navigation Applications* [online]. Dostupné z: http://www.bdtic.com/DownLoad/ADI/AN-602.pdf

[2] Sheu, J.S.; Huang, G.S.; Jheng, W.C.; Hsiao, C.H. *Design and Implementation of a Three-Dimensional Pedometer Accumulating Walking or Jogging Motions.* In Proceedings of the 2014 International Symposium on Computer, Consumer and Control (IS3C), Taichuang, Taiwan, 10–12 June 2014; pp. 828–831.

[3] *FRDM-K64F* [online]. [cit. 2019-04-01]. Dostupné z: https://os.mbed.com/platforms/FRDM-K64F/

[4] *FXOS8700CQ Data sheet* [online]. Dostupné z: https://www.nxp.com/docs/en/data-sheet/FXOS8700CQ.pdf

[5] *MMA8452Q Data sheet* [online]. Dostupné z: https://www.nxp.com/docs/en/data-sheet/MMA8452Q.pdf

[6] *ADXL345 Data sheet* [online]. Dostupné z: https://www.analog.com/media/en/technical-documentation/data-sheets/ADXL345.pdf

[7] *LSM303DLH Data sheet* [online]. Dostupné z: https://www.sparkfun.com/datasheets/Sensors/Magneto/LSM303%20Datasheet.pdf

[8] *FRDM-K64F Schematic* [online]. Dostupné z: https://www.nxp.com/webapp/Download?colCode=FRDM-K64F_SCH

[9] *SPARKFUN ELECTRONICS INC. BOB-13926 Schematic* [online]. Dostupné z: https://cdn.sparkfun.com/datasheets/Sensors/Accelerometers/SparkFun_MMA8452Q-Breakout.pdf

[10] *DFROBOT SEN0032 Schematic* [online]. Dostupné z: http://image.dfrobot.com/image/data/SEN0032/ADXL345_2.0%20sch.pdf

[11] *DFROBOT SEN0079 Schematic* [online]. Dostupné z: https://github.com/leffhub/Storage4Share/blob/master/LSM303%20Breakout%20Board%20-%20Tilt%20Compensated%20Compass%20SEN0079%20SCH.pdf

[12] *WAVE PCM soundfile format* [online]. Dostupné z: http://soundfile.sapp.org/doc/WaveFormat/

[13] *I^2C-bus specification and user manual* [online]. Dostupné z: https://www.nxp.com/docs/en/user-guide/UM10204.pdf

[14] *FatFs Module Application Note* [online]. [cit. 2019-04-01]. Dostupné z: `http://elm-chan.org/fsw/ff/doc/appnote.html`

[15] SMITH, Steven W. *The Scientist & Engineer's Guide to Digital Signal Processing*. California: Analog Devices, 1999, s. 261-350. ISBN 0-9660176-7-6.

[16] ZHAO, Neil. *Full-Featured Pedometer Design Realized with 3-Axis Digital Accelerometer* [online]. 2010 [cit. 2019-04-01]. Dostupné z: `https://www.analog.com/en/analog-dialogue/articles/pedometer-design-3-axis-digital-acceler.html`

A Obsah elektronické přílohy

- Aplikace pro vývojovou desku NXP FRDM-K64F.
- Měření provedená při testování.
- Programy pro testování filtrů a získání distribuce šumu.
- Text práce.